Joint Neural Networks for One-shot Object Recognition and Detection

Camilo Jose Vargas Cortes

PhD thesis

School of Electronic Engineering and Computer Science
Queen Mary University of London

2020

Abstract

This thesis presents a study of techniques for one-shot recognition and detection of objects. In computer vision, the one-shot object recognition task aims to categorize matches and mismatches between patterns from which a single sample is available for training. Whereas, the one-shot object detection task additionally defines a bounding box locating the position of the matching pattern within a target image. Such task is trivial for humans but challenging for machines.

Classical object recognition and detection have drawn a considerable amount of attention in the last decades leading to the development of several machine learning related approaches. Most approaches require a substantial amount of data to achieve state-of-art performance; making them unsuitable for use cases where collecting the data can be more costly than performing the task manually. In contrast, a relatively low amount of work is present in the literature tackling the one-shot recognition problem. Just until recently, Siamese Neural Networks have emerged in the literature as a deep learning approach to address the one-shot object recognition and detection problems while achieving reasonable results

This research focuses on building modifications to pairwise Siamese Neural Networks introducing the concept of joint layer, that improve their performance when addressing the one-shot object recognition problem in realistic scenarios. The proposed Joint Neural Networks achieved an accuracy of 70.0% when compared with the standard Siamese Neural Networks approach, outperforming the later by 10.0% accuracy tested on the MiniImageNet and QMUL-OpenLogo datasets.

Subsequently, a novel approach for one-shot object detection is presented as an extension of the Joint Neural Networks. The proposed one-shot detection approach is inspired by state-of-art one-stage detection approaches and does not rely on contextual information or support test sets to generate predictions. This approach shows competitive results, achieving a 56.1% mAP on the Pascal VOC

dataset split for one-shot detection. In adition, Joint Neural Networks push the state-of-art when trained on the COCO dataset and tested on the pascal VOC dataset, achieving 47.1% mAP.

Acknowledgements

I am very grateful to my family, friends and colleagues that directly or indirectly supported this work.

Statement of Originality

I, Camilo Jose Vargas Cortes, declare that to the best of my knowledge the content of this thesis is my own work, except where specific reference is made to the work of others.

the contents of this thesis are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

Camilo Jose Vargas Cortes

Signature: Date: 05th of April, 2020

Papers published while working on this thesis are listed in the first chapter of the thesis. Any publications produced in collaboration with others are clearly mentioned.

Contents

List of Figures

List of Tables

List of abbreviations

BCE	Binary cross entropy
BRIEF	Binary robust independent elementary feature
CNN	Convolutional neural network
DNN	Deep neural network
DROP	Distinctive region or pattern
FAST	eaturesfrom Accelerated Segment
HOG	Histogram Of Gradients
IOU	Intersection over union
JNN	Joint neural networks
LARKs	Locally adaptive regression kernels
MALM	Model-agnostic metalearning for fast adaptation of deep networks
ML	Machine learning
MTL	Meta-transfer learning for few-shot learning
MPEG	Moving picture expert group
ORB	Oriented FAST and Rotated BRIEF
RCNN	Region convolutional neural network
RPN	Region proposal network
SIFT	Scale Invariant Feature Transform
SNAIL	simple neural attentive meta-learner
SNN	Siamese neural network
SSD	Single shot multiBox detector
SURF	Speeded up robust features
TADAM	Task dependent adaptive metric for improved few-shot learning
YOLO	You only look once

Chapter 1

Introduction

1.1 Motivation

The automatic recognition, description, classification, and grouping of patterns are relevant tasks in a variety of engineering and scientific disciplines such as biology, psychology, medicine, marketing, security, transport, and remote sensing. The capability to generalize visual concepts from small amounts of data is a desired feature in modern computer vision systems where the data collection task can be highly costly. Regardless of almost 50 years of research, the design of a general-purpose method for pattern recognition remains an elusive goal [1].

Particularly in the security domain, government agencies rely on image and video data to recreate an event narrative to assist law enforcement and emergency relief. In this case, the information about objects or regions of interest is obtained manually by using a large number of audio-visual capturing devices including CCTV surveillance systems, content captured by broadcasters and content captured by mobile devices distributed on social networks [4]. In such cases the effort required to gather enough data to create a model that effectively recognize a suspicious individual or regions of interest is comparable to performing the task manually.

That is the case faced in the EU H2020 Lasie project [2], proposed to increase the efficiency of current investigation practices by providing a semi-automated

Figure 1.1: Sample frame from the Lasie dataset.

initial analysis of the vast amounts of heterogeneous forensic media data. The Video Analysis for Offender Identification use-case within the Lasie project is of particular interest for this research. Where, an user-selected distinctive region or pattern (DROP) is used to automatically detect corresponding regions in images or video frames from surveillance data to facilitate identification of a suspects. In other words, the user will select a distinctive pattern as a query (e.g. that a suspect has in his/her jacket) in a specific image, and the DROP module will search within a video database for frames where the same pattern appears. Figure 1.1 shows two sample frames from different videos for the DROP use-case taken from the Lasie dataset where the target region is the black jacket with an orange zipper.

Deep Neural Networks (DNNs) have achieved the state-of-art performance on a variety of pattern recognition tasks, most notably visual classification problems [6]. Typically, a very large amount of samples is required to obtain state-of-art results. Realistic use cases often require to achieve the defined goals with very limited datasets; in those cases deep neural networks performance falls short, overfitting on the training sets and producing poor generalization on the test sets [7].

Given that a good enough training dataset in terms of quality and size cannot be assumed available for building the models that can recognize and localize regions of interest, the definition of reliable methods and standards to automatically perform pattern recognition/detection without relying on training data become paramount. Despite the notorious growth of the pattern recognition

field, relatively low efforts have been made to study this problem regardless of its relevance [5]. Such approaches can be generalized to more common uses cases like the solution presented by google lens [3]; that allows an end user to perform a data base search of products using a single picture of the object of interest.

Motivated by the aforementioned needs, this work focuses on the one-shot recognition and detection problem. The main challenge of this research is to propose a method to automatically recognize/detect objects under one-shot constraints for heterogeneous media contents.

1.2 Problem definition

Classical approaches for one-shot recognition assign categories to an input image from a set of fixed categories learned using training data. Given that such data is not always available in real world applications, the main goal of this research is to develop a one-shot recognition and detection approach that re-maps the traditional object detection and recognition problem allowing class definition on the fly as an input of the proposed model. Hence, the proposed model will be able to perform under circumstances where no training data is available for particular classes.

Let $\{(x_i, y_i), i = 1, ..., n\}$ be a set of images x_i with labels $y_i \in \{1, 2, ..., C\}$ for a number of training samples n and classes C. Then a training set can be built as follows $\mathbb{S}_{train} = \{(x_i, x_j, Z_{ij}), i, j = 1, ..., n\}$, where $Z_{ij} = \{0, y_i = y_j; 1, y_i \neq y_j\}$. Similarly a testing set can be constructed, where $\{(x'_i, y'_i), i' = 1, ..., n'\}$ is a set of images x'_i with labels $y'_i \in \{1, 2, ..., C'\}$ for a number of testing samples n' and classes C', and the testing set is given by $\mathbb{S}_{test} = \{(x'_i, x'_j, Z'_i j), i, j = 1, ..., n\}$, where $Z'_i j = \{0, y'_i = y'_j; 1, y'_i \neq y'_j\}$. In particular the label indexes from y and y' do not correspond to the same categories; in other words, the training and testing datasets do not overlap in terms of categories.

To simplify, let $x^{(1)}$, $x^{(2)}$ be the query and target images during either training or testing inferred by context. The recognition problem is formulated as building a model $\mathbb{M}(x^{(1)}, x^{(2)})$ using the training set $\mathbb{S}_{train}$ in order to perform

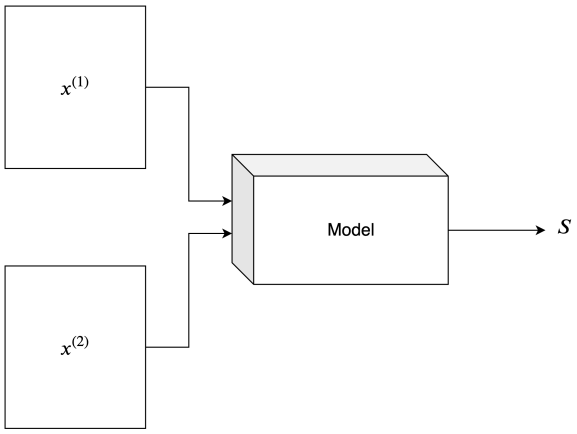


Figure 1.2: Recognition problem overview.

binary classification on the test dataset $\mathbb{S}_{test}$ defining matches or mismatches between the input images according to the visual patterns represented by the labels y and y'. Hence, any proposed solution must accept a query $x^{(1)}$ and target $x^{(2)}$ images as input and deliver an output S that defines a match or mismatch between the patterns in $x^{(1)}$, $x^{(2)}$. The model output S can be given either by a thresholded similarity metric computed over the input images or a straight binary output representing the categories match or mismatch. Figure 1.2 shows an overview of the problem structure for the one-shot recognition case.

The one-shot detection problem shares a similar formalization and structure as the one presented previously for recognition. A query and target input images $x^{(1)}$, $x^{(2)}$ are processed by a model $\mathbb{M}'(x^{(1)}, x^{(2)})$ to output a vector S that will contain a confidence level and a set of coordinates specifying the location of the query pattern via bounding boxes. This follows approaches such as [8] and [9].

Again, this research focuses on the case where models $\mathbb{M}(x^{(1)}, x^{(2)})$, $\mathbb{M}'(x^{(1)}, x^{(2)})$ do not contain any information about the test categories assessed for the input query $x^{(1)}$, which are called unseen categories; in other words, categories that are not used during the learning stages. The main reason for focusing in the specific case of unseen classes is derived from the application level, where there is a single sample of the query image category and the pattern recognition/detection is expected to be as quick as possible. As a consequence, additional training or fine-tuning stages are avoided, since that would only delay further the search

process. Advances in this topic will impact different application fields by providing tools to perform generalized retrieval and matching of patterns for which a single sample is available.

1.3 Research objectives

- **O1:** Establish the state-of-art approaches available to engage with the One-shot object detection task. This objective is addressed by the literature review presented in chapter 2.

- **O2:** Propose an approach to adequate and simplify the use of the Viola-Jones algorithm for one shot object detection. Chapter 3 introduces a Viola-Jones algorithm modification along with an assessment of image processing based approaches for one-shot object detection. Results open the door to explore more sophisticated techniques to approach the one-shot object detection task.

- **O3:** Analyse the performance of Siamese Neural Networks for one-shot object detection in a realistic scenario. Given the success of Siamese Neural Networks, chapter 4 evaluated their performance under different configurations for one-shot logo recognition.

- **O4:** Introduce an strategy to improve recognition results of the Siamese Neural Networks for one-shot object recognition. In consequence, chapter 5 presents a novel Convolutional Neural Network architecture called Joint Neural Networks. The proposed approach is based on Siamese Neural Networks and leverages the data sharing between branches using the proposed joint layers.

- **O5:** Extend the proposed Joint Neural Networks for one-shot object detection. Chapter 5 in its section 5.3 introduces an approach to perform one-shot object detection, developed from the initially proposed Joint Neural Networks for one-shot recognition.

1.4 Research methodology

Computer vision has emerged in recent years as a feasible alternative to deal with many real world problems creating a relatively new demand to build modern systems that tackle the drawbacks of traditional machine learning techniques such as data volume dependency. For this reason, a literature review is performed to examine any set of techniques that can support solving this challenging problem. The keywords for relevant research in this topics are:

- One-shot object detection/recognition
- One-shot Siamese neural networks
- One-shot learning

Papers are reviewed quantitatively to check the one-shot performance in well known datasets. Additionally, the key points of the proposed approaches are assessed; to either take useful concepts of the proposed techniques or to establish the trends that constitute the state-of-art. This process defines the base of knowledge used to develop the proposed approach and carry on with the experimental sections.

From the resulting literature review the research works suitable to address the one-shot object recognition and detection problems are categorized (sections 2.1, 2.2.1). A study on a real world application of the image processing based approaches for one-shot object detection is carried out in chapter 3 in order to assess their performance.

After identifying the drawbacks of image processing techniques applied to the one-shot recognition and detection problem, a further study is performed using the dominant approach in the literature for one-shot object recognition: Siamese neural networks (chapter 4). This study exhibits a potential point of improvement for pairwise models that is later explored in chapter 5.

Finally, in consequence with the concept presented in section 5.1 and inspired by state-of-art one-stage object detection approaches, chapter 5 introduces an adaptation of the so called Joint Neural Networks to perform one-stage object detection under the one-shot constraints.

1.5 Thesis structure

Chapter 2 Presents a description of the literature review that defines the base knowledge for the proposed approach and related experiments.

Chapter 3 Explores the performance of image processing approaches tackling the one-shot detection problem.

Chapter 4 Applies and adapts the Siamese Neural Networks approach for the one-shot recognition task addressed in this research.

Chapter 5 Introduces the Joint layers, which is the fundamental concept of the proposed approach to one-shot object recognition. In addition the proposed Joint Neural Networks are adapted for one-shot object detection inspired by concepts of mainstream state-of-art object detection approaches.

1.6 Contributions and associated publications

- Chapter 3: Presents an evaluation of image processing techniques for one-shot detection in the wild. The study assesses the performance of image processing based techniques for one-shot detection in the context of surveillance, using the Lasie dataset introduced for the Video Analysis for Offender Identification use case. This work was published as part of the paper "Ontology Population Framework of MAGNETO for Instantiating Heterogeneous Forensic Data Modalities", theorizing the use of distinctive region or pattern detection to assist forensic investigations. Presented at the IFIP: International Conference on Artificial Intelligence Applications and Innovations, by Ernst-Josef Behmer, Krishna Chandramouli, Victor Garrido, Dirk Mhlenberg, Dennis Mller, Wilmuth Mller, Dirk Pallmer, Francisco J. Prez, Tomas Piatrik, Camilo Vargas, AIAI 2019. In addition, this work was also presented at the security event IFSEC 2017 as part of the Lasie project.

- Chapter 4: Introduces a performance assessment study based on Siamese neural networks for one-shot object recognition involving the use of differ-

ent Deep neural networks as the backbone of the architecture. Commonly in the literature the AlexNet architecture is selected by default to be the backbone of SNNs architecture. The presented study highlights the impact that different embedded architectures have over the one-shot object recognition problem on the Open-Logo dataset. This study is published as "One Shot Logo Recognition Based on Siamese Neural Networks" in the ACM International Conference on Multimedia Retrieval (ICMR 2020) by Camilo Vargas, Qianni Zhang, Ebroul Izquierdo.

- Chapter 5: Presents a scalable architecture akin to the standard Siamese architecture to undertake the one-shot recognition and detection problems. By introducing the Joint layer concept the traditional Siamese neural network approach is adapted to leverage communication between the sister networks and improve recognition results. Additionally, and inspired by state-of-art one-stage detection methods, the so called Joint Neural Networks initially introduced for one-shot object recognition are modified to address one-shot object detection task. This work is being prepared for a Journal submission as "Joint Neural Networks for One-shot Object Recognition and Detection".

Chapter 2

Theoretical Framework and Background

Computer vision plays a fundamental role in an ample number of application fields where visual object recognition and detection are some of the crucial tasks. In consequence, in the last few decades computer vision research has gained considerable momentum; new methods and datasets to approach the aforementioned visual tasks in specific use cases have been proposed leading to the development and integration of computer vision systems in industry.

Most of modern computer vision systems rely on machine learning approaches that require a high amount of training data to build the backbone models that support the automated visual tasks. High accuracy is an expected feature in state-of-art computer vision methods within specialized applications. In contrast, the data requirement that supports the construction of accurate models, often becomes one of the main limitations when developing automated visual solutions in most application cases. This is where approaches that rely on little data to generate stable model become relevant.

This chapter explores techniques available in the state-of-art literature for one-shot object detection and recognition. First, early techniques based on image processing approaches are reviewed. Second, an introduction to the standard object detection state-of-art is presented, reviewing in detail the "You

Only Look Oonce" approach which is of particular importance for this research. Then, approaches for object recognition under small data constraints are examined; including zero-shot, one-shot and few-shot learning approaches. Finally, the datasets, evaluation metrics, and tools used for the experimental development of this research are discussed. By the end of the chapter the required background to approach chapters 3 to 5 is established.

2.1 Image processing approaches for one-shot object recognition and detection

Previous to modern machine learning approaches, the initial attempts to engage in the detection and recognition problems were based on image processing. Many of the image processing based algorithms for one-shot object detection/recognition found in the literature rely on the location and description of interest points or keypoints. Commonly, a set of points of interest or keypoints is computed for the query and target images. Then, a feature extraction process is performed in order to describe each one of the detected keypoints in the query and target images. Finally, a matching stage takes place where the keypoint features of each image is compared in order to define where the query object may be located in the target image in the detection case. Approaches based on keypoint detection and description often offer feature scale, rotational, and illumination invariance.

The Scale Invariant Feature Transform (SIFT) developed by Lowe [10][11] is a well known method for finding interest points and feature descriptors, providing invariance to scale, rotation, illumination, affine distortion, perspective and similarity transforms, and noise. Lowe demonstrates that by using several SIFT descriptors together to describe an object, there is additional invariance to occlusion and clutter, since if a few descriptors are occluded others will be found [10].

SIFT is a complete algorithm and processing pipeline, including both an interest point and a feature descriptor method. SIFT includes stages for selecting center-surrounding circular weighted Difference of Gaussian (DoG) maxima

interest points in scale space to create scale-invariant keypoints (a major innovation). The feature extraction step involves calculating a binned Histogram Of Gradients (HOG) structure from local gradient magnitudes into Cartesian rectangular bins, or into log polar bins using the GLOH variation, at selected locations centered around the maximal response interest points derived over several scales.

The descriptors are fed into a matching pipeline to find the nearest distance ratio metric between the closest match and second closest match, which considers a primary match and a secondary match together and rejects both matches if they are too similar, assuming that one or the other may be a false match. The local gradient magnitudes are weighted by a strength value proportional to the pyramid scale level, and then binned into the local histograms. In summary, SIFT is a very well thought out and carefully designed multi-scale localized feature descriptor. The Speeded-up Robust Features Method (SURF) [12] described in detail in section 2.1.1 is inspired from SIFT. It operates in a scale space and uses a fast Hessian detector based on the determinant maximum points of the Hessian matrix.

The Binary robust independent elementary feature (BRIEF) [13] [14] descriptor uses a random distribution pattern of 256 point-pairs in a local 31x31 region for the binary comparison to create the descriptor. One key idea with BRIEF is to select random pairs of points within the local region for comparison. BRIEF is a local binary descriptor and has achieved very good accuracy and performance in robotics applications [15]. BRIEF and ORB are closely related, although the ORB point-pair pattern descriptor is also built differently.

The Oriented FAST and Rotated BRIEF (ORB) [16] is an acronym for Oriented BRIEF. As the name suggests, ORB is based on BRIEF adding rotational invariance. This is achieved by determining corner orientation using Features from Accelerated Segments (FAST) [17], followed by a Harris corner metric to sort the keypoints. The corner orientation is refined by intensity centroids using Rosins method [18]. The FAST, Harris, and Rosin processing is performed at each level of an image pyramid scaled with a factor of 1.4, rather than the common octave pyramid scale methods. It should be noted that ORB is a highly

optimized and very well engineered descriptor, with careful considerations on computational speed, memory footprint, and accuracy. Many of the descriptors surveyed in this section are primarily research projects, with less priority given to practical issues, but ORB focuses on optimizing and practical issues.

The Daisy Descriptor [19] [20] is inspired by SIFT and GLOH-like descriptors, and is devised for dense-matching applications such as stereo mapping and tracking, reported to be about 40 percent faster than SIFT. Daisy relies on a set of radially distributed and increasing sized Gaussian convolution kernels that overlap and resemble a flower-like shape (Daisy). Daisy does not need local interest points, and instead computes a descriptor densely at each pixel, since the intended application is stereo mapping and tracking. Rather than using gradient magnitude and direction calculations like SIFT and GLOH, Daisy computes a set of convolved orientation maps based on a set of oriented derivatives of Gaussian filters to create eight orientation maps spaced at equal angles.

In contrast, some image processing based algorithms for object detection define a similarity measurement to compare images or image regions. Such is the case of the early template matching approaches of LARKs [21], where the correlation between a region in the target image and the query pattern is measured at different locations and sizes. Usually, the kernel based approaches are slower when compared to keypoint approaches mainly because the detection process for kernel based approaches evaluate different locations and region sizes in the target image. In particular, the SURF and LARKs approaches used for comparison purposes in chapter 3 are discussed in the following subsections.

2.1.1 Speeded up robust features

The speeded up robust features (SURF) is a local feature detector and descriptor that focuses on bloblike structures in the image. These structures can be found at corners of objects, as well as at locations where the reflection of light on specular surfaces is maximal.

This approach for interest point detection uses a Hessian matrix approximation. This leads to the use of integral images as made popular by Viola and

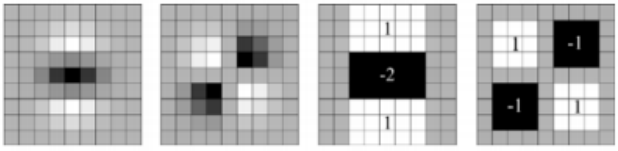


Figure 2.1: Left to right: The (discretized and cropped) Gaussian second order partial derivative in y and xy directions respectively; the SURF approximation for the second order Gaussian partial derivative in y and xy directions respectively. The grey regions are equal to zero [12]

Jones, which reduce the computation time drastically [86]. Integral images fit in the more general framework of boxlets, as proposed by [12].

Given a point $\mathbf{x} = (x, y)$ in an image I, the Hessian matrix $H(\mathbf{x}, \sigma)$ in $\mathbf{x}$ at scale σ is defined as follows:

$$H(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{xy}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix}, \tag{2.1}$$

where $L_{xx}(\mathbf{x}, \sigma)$ is the convolution of the Gaussian second order derivative $\frac{\partial}{\partial x^2} g(\sigma)$ for the image I at the point $\mathbf{x}$, and similarly for $L_{xy}(\mathbf{x}, \sigma)$ and $L_{yy}(\mathbf{x}, \sigma)$.

Gaussians are optimal for scale-space analysis but in practice they have to be discretized and cropped (Figure 2.1, left half). This leads to a loss in repeatability under image rotations around odd multiples of $\frac{\pi}{4}$. This weakness holds for Hessian-based detectors in general.

Nevertheless, the approximated detectors still perform well and the slight decrease in performance does not outweigh the advantage of fast convolutions brought by the discretization. This approach pushes the approximation for the Hessian matrix with box filters (right half of Figure 2.1). These approximate second order Gaussian derivatives and can be evaluated at a very low computational cost using integral images. The calculation time therefore is independent of the filter size.

The 9×9 box filters in Figure 2.1 are approximations of a Gaussian with $\sigma = 1.2$ and represent the lowest scale for computing the blob response maps, denoted by D_{xx}, D_{yy}, and D_{xy}. The weights applied to the rectangular regions

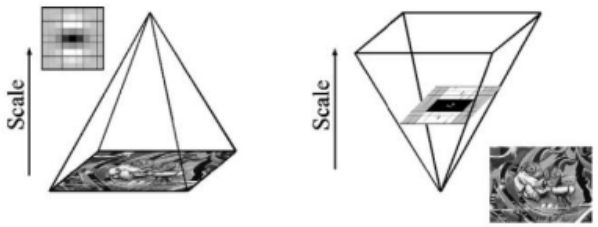


Figure 2.2: Instead of iteratively reducing the image size (left), the use of integral images allows the up-scaling of the filter at constant cost (right) [12]

are kept simple for computational efficiency. This yields to:

$$det(H_{approx}) = D_{xx}D_{yy} - (wD_{xy})^2. \tag{2.2}$$

The relative weight w of the filter responses is used to balance the expression for the Hessians determinant. This is needed for the energy conservation between the Gaussian kernels and the approximated Gaussian kernels,

$$w = \frac{|L_{xy}(1,2)|_F|D_{yy}(9)|F}{|L_{yy}(1,2)|_F|D_{xy}(9)|F} \approx 0.9, \tag{2.3}$$

where $|x|_F$ is the Frobenius norm. Notice that for theoretical correctness, the weighting changes depending on the scale.

The scale space is analysed by up-scaling the filter size rather than iteratively reducing the image size, as shown in Figure 2.2. The scale space is divided into octaves that represent the filter response maps obtained by convolving the same input image with a filter of increasing size. Each octave is subdivided into a constant number of scale levels.

SURF describes the distribution of the intensity content within the interest point neighbourhood, similar to the gradient information extracted by SIFT [11]. The SURF descriptor is built based on the distribution of first order Haar wavelet responses in x and y direction rather than the gradient, exploiting integral images for speed, and uses 64 dimensional descriptor reducing the feature

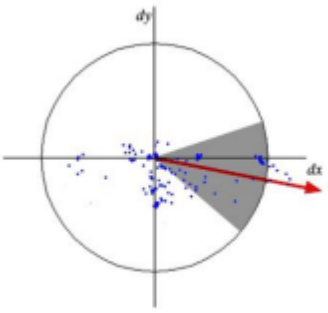


Figure 2.3: Orientation assignment: a sliding orientation window detects the dominant orientation of the Gaussian weighted Haar wavelet responses at every sample point within a circular neighborhood around the interest point [12]

computation and matching computation times.

In order to be invariant to image rotation, the Haar wavelet responses in and direction within a circular neighbourhood around the interest point is calculated using integral images for fast filtering. After computing the wavelet responses, they are represented as points in a space with the horizontal response strength along the abscissa and the vertical response strength along the ordinate. The dominant orientation is estimated by calculating the sum of all responses within a sliding orientation window, see Figure 2.3. The horizontal and vertical responses within the window are summed. The two summed responses then yield a local orientation vector. The longest such vector over all windows defines the orientation of the interest point.

The first step for descriptor computation consists of constructing a square region centered around the interest point and oriented along the orientation selected in previous section. The region is split up regularly into smaller 4×4 square sub-regions as illustrated in Figure 2.4, preserving important spatial information. For each sub-region, the Haar wavelet responses at 5×5 regularly spaced sample points is computed. Here, d_x and d_y are the Haar wavelet response in horizontal and vertical direction respectively. Then, the wavelet responses d_x and d_y are summed up over each sub-region in order to form a first set of entries in the feature vector. In order to bring in information about the polarity of the intensity changes, the sum of the absolute values of the responses $|d_x|$ and $|d_x|$ is also extracted. Hence, each sub-region has a 4D descriptor vector

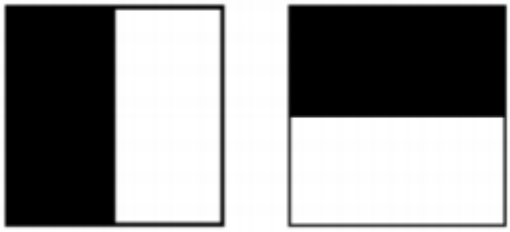


Figure 2.4: Haar wavelet filters to compute the responses in x (left) and y direction (right). The dark region have a weight equals to 1 and the light regions to -1 [12].

for its underlying intensity structure $v = (\Sigma d_x, \Sigma d_y, \Sigma|d_x|, \Sigma|d_y|)$.

Once the descriptor vector v is computed, the OpenCV library uses the Fast Approximate Nearest Neighbor Search (FLANN) to match the descriptors for each keypoint in the query and target images and later thresholds the distance between descriptors and the amount of key-points matched to define a detection.

2.1.2 Locally adaptive regression kernels

The Locally adaptive regression kernels (LARKs) presented in [21] show an algorithm capable of searching for a visual object of interest without training. This approach operates using a single example of an object of interest to find similar matches from which does not require prior knowledge or preprocessing steps. This method is based on the computation of local regression kernels as descriptors from a query, which measure the likeness of a pixel to its surroundings. Salient features are extracted from said descriptors and compared against analogous features from the target image. The comparison is performed using a matrix generalization of the cosine similarity measure. Figure 2.5 shows the architecture of the LARKS approach.

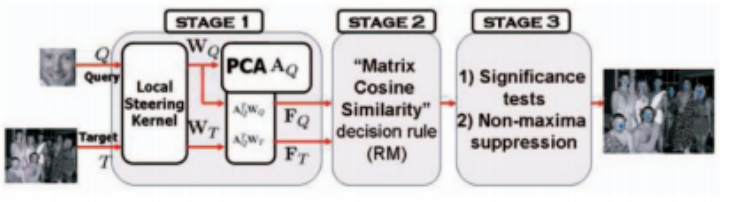


Figure 2.5: LARKs architecture overview [21].

The fundamental component of this approach is the calculation of the local steering kernel (LSK), which essentially measures the local similarity of a pixel to its neighbors both geometrically and photometrically. The key idea is to robustly obtain local data structures by analyzing the photometric differences based on estimated gradients and use this structure information to determine the shape and size of a canonical kernel. Denoting the target image T and the query image Q, a dense set of local steering kernels is computed from each image. These densely computed descriptors tend to be over-complete (redundant) Therefore, the PCA dimensionality reduction approach is applied to the resulting feature arrays. Generally the target image T, is bigger than the query image Q. Hence, the target image is divided into a set of overlapping patches which are the same size as Q.

This work uses Matrix Cosine as a measure of similarity to compare and match the feature vectors. Figure 2.6 shows the results of this method for two different query patterns.

To determine the shape and size of a canonical kernel. The local kernel is modeled as a radially symmetric function:

$$K(x_l - x; H_l) = \frac{K(H_l^{-1}(x_l - x))}{det(H_l)}, l = 1, ..., P^2, \tag{2.4}$$

where $x_l = [x_1, x_2]_l^T$ are the spatial coordinates, P^2 is the number of pixels in a local window $(P \times P)$, and the steering matrix is defined as:

$$H_l = hC_l^{-\frac{1}{2}} \in \mathbb{R}^{(2\times 2)}, \tag{2.5}$$

where is h a global smoothing parameter and the matrix C_l is a covariance matrix estimated from a collection of spatial gradient vectors within the local analysis window around a position x. The steering matrix H_l modifies the shape and size of the local kernel in a way that roughly encodes the local geometric structures present in the image. With such steering matrices, a Gaussian func-

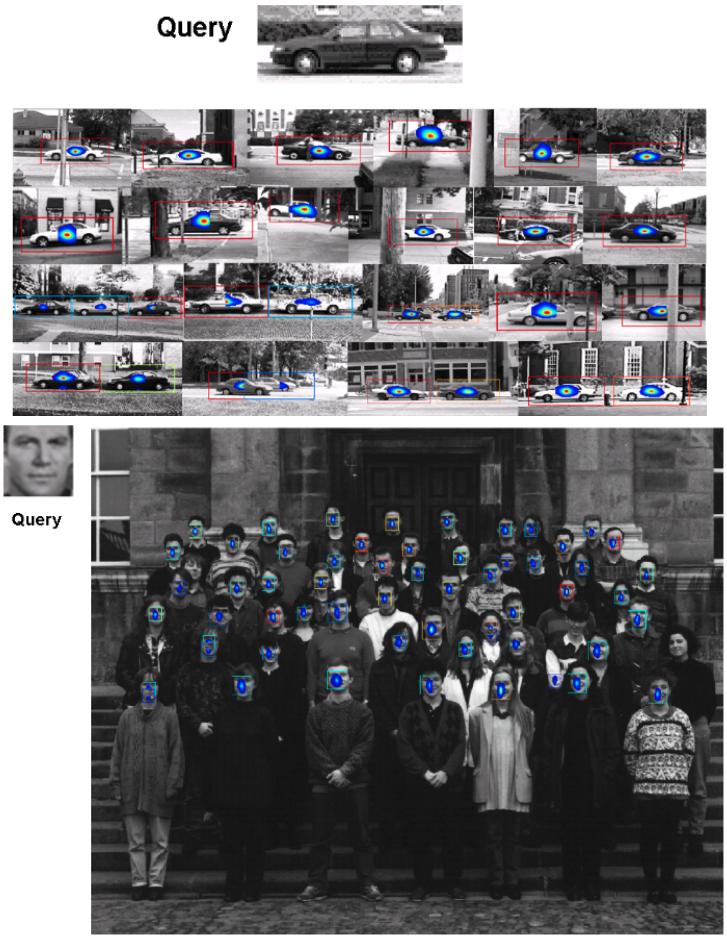


Figure 2.6: Two different test results using LARKs [21]

tion is chosen for K, which leads to the following form for the local steering kernels (LSKs):

$$K(x_l - x : H_l) = \frac{\sqrt{det(C_l)}}{2\pi h^2} exp\Big\{ - \frac{(x_l - x)^T C_l (x_l - x)}{2h^2} \Big\}. \tag{2.6}$$

At a position x, essentially a normalized version of the function $K^j(x_l - x; H_l)$ is used to represent an images inherent local geometry. To be more specific, the local steering kernel function $K^j(x_l - x; H_l)$ at a patch indexed by j is densely calculated and normalized as follows:

$$\begin{aligned} W_Q^j(x_l - x) &= \frac{K_Q^j(x_l - x; H_l)}{\sum_{i=1}^{P^2} K_Q^j(x_l - x; H_l)}, \begin{cases} j = 1, ..., n \\ l = 1, ..., P^2 \end{cases} \\ W_T^j(x_l - x) &= \frac{K_T^j(x_l - x; H_l)}{\sum_{i=1}^{P^2} K_T^j(x_l - x; H_l)}, \begin{cases} j = 1, ..., n_T \\ l = 1, ..., P^2 \end{cases} \end{aligned}, \tag{2.7}$$

where n and n_T are the number of patches where LSKs is computed in the query image Q and the target image T respectively. In order to organize $W_Q^j(x_l - x)$ and $W_T^j(x_l - x)$, which are densely computed from Q and T, let W_Q, W_T be matrices whose columns are vectors w_Q^j, w_T^j which are column stacked versions of $W_Q^j(x_l - x)$ and $W_T^j(x_l - x)$ respectively:

$$\begin{aligned} W_Q &= [w_Q^1, ..., w_Q^n] \in \mathbb{R}^{(P^2 \times 2)} \\ W_T &= [w_T^1, ..., w_T^n] \in \mathbb{R}^{(P^2 \times n_T)} \end{aligned} \tag{2.8}$$

As described in Figure 2.5, the next step is to apply PCA to W_Q for dimensionality reduction and to retain only its salient characteristics. By applying PCA to W_Q, the first d principal components are retained, which form the columns of a matrix $A_Q \in \mathbb{R}^{(P^2 \times d)}$. Next, the lower dimensional features are computed by projecting W_Q and W_T onto A_Q:

$$F_Q = [f_Q^1, ..., f_Q^n] = A_Q^T W_Q \in \mathbb{R}^{(d \times n)}$$
$$F_T = [f_T^1, ..., f_T^n] = A_Q^T W_T \in \mathbb{R}^{(d \times n_T)} \tag{2.9}$$

2.2 Machine learning approaches for one-shot recognition and detection

As the performance of hand-crafted features derived from image processing approaches became saturated, object detection reached a plateau [22]. This slow progress stage was interrupted with the rise of deep Convolutional Neural Networks which are able to learn robust and high-level feature representations of an image [23].

Convolutional Neural Networks (CNNs) are a well-known deep learning architecture inspired by the natural visual perception mechanism of the living creatures. It can obtain effective representations of the original image, which makes it possible to recognize visual patterns directly from raw pixels with little-to-none preprocessing [24]. Convolutional Neural Networks have been extensively studied, triggering a rapid performance improvement in the automation of various visual tasks and challenges. Furthermore, CNNs triggered the expansion of annotated training and testing datasets for different tasks/applications, as well as great improvements of graphics processor units that support current state-of-art results.

The following subsection briefly introduce the history of recent mainstream methods for object detection and recognition based on convolutional Neural Networks that require considerable amounts of data to perform adequately. The "You Only Look Once" Method [8] is explored in detail given its relevance to chapter 5.

2.2.1 Deep learning for object detection and recognition

Initially used for image recognition, convolutional networks were quickly adapted for classification with localization and later for multi-box object detection tasks.

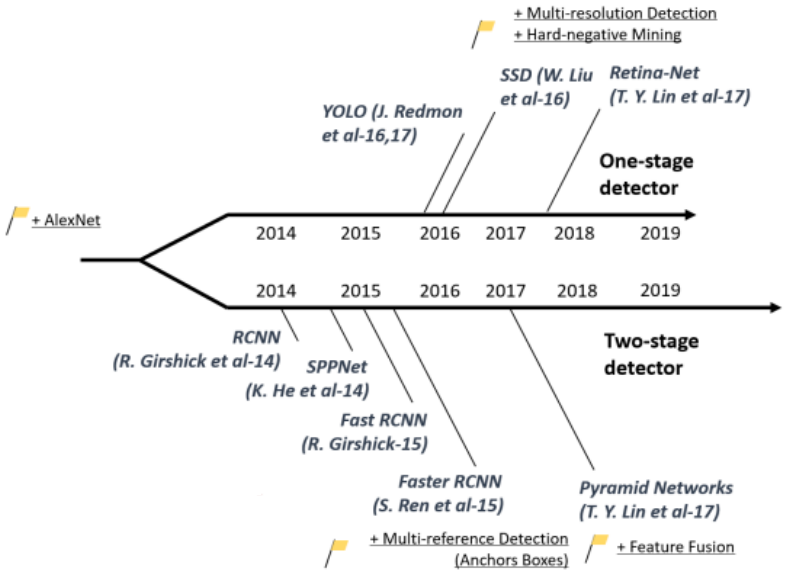


Figure 2.7: A road map of deep learning based object detection approaches taken from [22]. Milestone detectors in this figure: R-CNN [25], SPPNet [26], Fast-RCNN [27], Faster-RCNN [28], YOLO [8], SSD [9], Pyramid Networks [29], Retina-Net [30].

Following the literature, object detection methods can be classified as one-stage or two-stage approaches. One-stage methods such as "You Only Look Once" (YOLO) [8] or "Single Shot MultiBox Detector" (SSD) [9], focus on producing multi-box detection results within a single process pipeline approach. In other words, they process the target image in a single pass as a whole to produce the multi-box detection. Whilst, two-stage approaches attempt to first generate region proposals to be subsequently classified independently.

Two-stage multi-box object detection approaches emerged first with the work presented in [31] and [25]. There, the authors proposed the selective search approach that adapted the segmentation method proposed by Felzenszwalb [32] to generate the region proposals. Afterwards each proposed region is rescaled to a fixed image size and fed into a CNN model to generate the region categories. Although Region-based Convolutional Neural Networks (R-CNN) made great progress, its main drawbacks are the redundant feature computations on a large number of overlapped proposals (over 2000 boxes from one image) leading to an extremely slow detection speed (14s per image with GPU) [22].

The fast Region-based Convolutional Neural Networks (Fast-RCNN) [27] detector was proposed as a further improvement from R-CNN. Fast-RCNN integrates the advantages of R-CNN and Spatial Pyramid Pooling networks (SPP-Net) to compute a convolutional feature map for the entire input image and then classify each object proposal using a feature vector extracted from the shared feature map, i.e. taking advantage of overlapping region proposals to speed-up computation times. Despite its notable progress the detection speed is still limited by the region proposal algorithm. On VOC07 dataset, Fast-RCNN increased the mean average presicion (mAP) from 58.5% (R-CNN) to 70.0% while with a detection speed over 200 times faster than R-CNN [22].

The Faster-RCNN detector [28] was proposed shortly after the Fast-RCNN. Faster-RCNN is the first end-to-end, and the first near-realtime deep learning detector (COCO mAP@.5=42.7%, COCO mAP@[.5,.95]=21.9%, VOC07 mAP=73.2%, VOC12 mAP=70.4%, 17fps). The main contribution of Faster-RCNN is the introduction of Region Proposal Network (RPN) that enables nearly cost-free region proposals. From R-CNN to Faster-RCNN, most individual blocks of an object detection system, e.g., proposal detection, feature extraction, bounding box regression, etc, have been gradually integrated into a unified, end-to-end learning framework. Although Faster RCNN breaks through the speed bottleneck of Fast-RCNN, there is still computation redundancy at the subsequent detection stage [22].

One-stage detection approaches emerged as a faster option to two-stage detectors. YOLO is the first one-stage detector in the deep learning era [8][22]. YOLO is extremely fast compared to two-stage approaches; the fast version of YOLO runs at 155fps with VOC07 mAP=52.7%, while its enhanced version runs at 45fps with VOC07 mAP=63.4% and VOC12 mAP=57.9% [22]. The YOLO approach applies a single neural network to the full image, dividing the input image into regions and predicting bounding boxes and probabilities for each region simultaneously. Further improvements in detection accuracy and detection speed are made to YOLO in its more recent versions [33][34].

The SSD method [9] was proposed the same year as YOLO. It is the second one-stage detector in the deep learning era. The main contribution of

SSD is the introduction of the multi-reference and multi-resolution detection techniques, which significantly improves the detection accuracy of a one-stage detector, especially for some small objects. SSD has advantages in terms of both detection speed and accuracy compared to the first version of YOLO (VOC07 mAP=76.8%, VOC12 mAP=74.9%, COCO mAP@.5=46.5%, mAP@[.5,.95]=26.8%, the fast version runs at 59fps). The anchor window approach described in SSD is adopted by YOLO approach in its subsequent versions.

The following subsection presents the technical details of YOLO and YOLO version 2. This encapsulates most of the concepts used in the proposed approach in chapter 5

You only look once

You Only Look Once is a state-of-art, real-time object detection approach developed around 2015. It outperformed every other object detection approach at that time. YOLO [8] unifies the region proposal and classification components of object detection into a single neural network, using features from the entire image to predict each bounding box. It also predicts all bounding boxes across all classes for an image simultaneously, i.e., the model reasons globally about the full image and all the objects in the image. The YOLO design enables end-to-end training and realtime speeds while maintaining high average precision.

The YOLO approach divides the input image into an $S \times S$ grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those boxes. These confidence scores reflect how confident the model is that the box contains an object and also how accurate it thinks the box is that it predicts. The confidence is defined as $Pr(Object) * IOU_{truth}^{pred}$. If no object exists in a particular cell, the confidence scores should be zero; Otherwise, the confidence score should be equal to the intersection over union (IOU) between the predicted box and the ground truth.

The YOLO output is modeled to represent 5 values per bounding box: x, y, w, h, and confidence. The (x, y) coordinates represent the center of the box

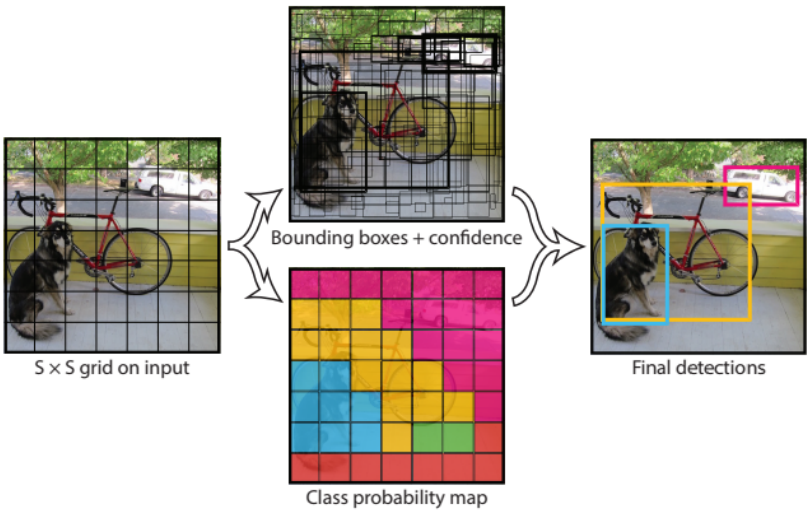


Figure 2.8: YOLO bounding box and class prediction overview, taken from [8]

relative to the bounds of the grid cell. The width and height are predicted relative to the whole image. Finally the confidence prediction represents the IOU between the predicted box and any ground truth box.

Additionally, each grid cell also predicts C conditional class probabilities, $Pr(Class_i|Object)$. These probabilities are conditioned on the grid cell containing an object. So the first version of YOLO is limited to one set of class probabilities per grid cell, regardless of the number of boxes B. Figure 2.8 shows an overview af the approach, the final output vector encodes predictions as an $S \times S \times (B * 5 + C)$. In particular the first version of YOLO is evaluated using $S = 7$, $B = 2$, and $C = 20$ (for the 20 PASCAL VOC classes); for a final $7 \times 7 \times 30$ prediction tensor.

At test time the conditional class probabilities and the individual box confidence predictions are multiplied to obtain class-specific confidence scores for each box as in eq 2.10.

$$Pr(Class_i|Object) * Pr(Object) * IOU_{truth}^{pred} = Pr(Class_i) * IOU_{truth}^{pred} \quad (2.10)$$

These scores encode both the probability of that class appearing in the box

and how well the predicted box fits the object.

The YOLO model architecture is defined by a Convolutional Neural Network where the initial convolutional layers are meant to extract features from the image while the fully connected layers predict the output probabilities and coordinates.

The network architecture is inspired by the GoogLeNet model for image classification [35], with 24 convolutional layers followed by 2 fully connected layers. Instead of the inception modules used by GoogLeNet, YOLO uses 1×1 reduction layers followed by 3×3 convolutional layers, the network architecture is displayed in Figure 2.9.

The YOLO loss function is defined as follows:

$$
\begin{aligned}
&\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \Big[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \Big] \\
&+ \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \Big[(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \Big] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \Big(c_i - \hat{c}_i \Big)^2 \\
&+ \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} \Big(c_i - \hat{c}_i \Big)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{ij}^{obj} \sum_{C \in classes} (p_i(C) - \hat{p}_i(C))^2,
\end{aligned}
\tag{2.11}
$$

where S is the number of cells in the grid used to divide the input image; B is the number of bounding boxes predicted in each cell; x, y, w, h, c are the four coordinates and confidence predicted for each bounding box; C is the class predicted per cell in the $S \times S$ grid; $\mathbb{1}_i^{obj}$ denotes if object appears in cell i and $\mathbb{1}_{ij}^{obj}$ denotes that the jth bounding box predictor in cell i is responsible for that prediction; and the user defined constants λ_{coord} and λ_{noobj} are set to balance how much cells not containing objects contribute to the loss ($\lambda_{coord} = 5, \lambda_{noobj} = 0.5$) [8].

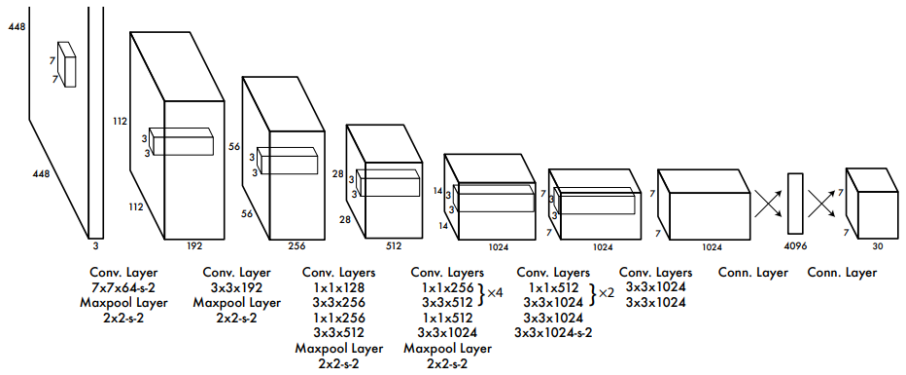


Figure 2.9: YOLO version 1 architecture [8]

The second version of YOLO [33] was proposed in 2016. Two main contibutions of YOLO v2 are of particular interest for this thesis. Firstly, the architecture of the model is updated to the called Darknet-19 and switched to convolutional layers in its whole pipeline. Table 2.1 shows the detailed layer description of the Darknet-19 used for classification. For detection purposes the network is modified by removing the last convolutional layer and three 3×3 convolutional layers with 1024 filters are added followed by a 1×1 convolutional layer with the number of outputs needed for detection. This means that when tested on the VOC dataset, 5 boxes with 5 coordinates each and 20 classes per box are predicted (125 filters).

Secondly, the second version of YOLO [33] takes the anchor boxes concept proposed in the single shot multibox detector approach [9]. Anchor boxes are a set of predefined box shapes that are expected to predict the location of objects for each cell. The idea behind the use of anchor boxes is to associate a set of default bounding boxes with each feature map cell in the $S \times S$ grid. The anchor boxes tile the feature map in a convolutional manner, so that the position of each box relative to its corresponding cell is fixed. The offsets relative to the anchor boxes in a cell, as well as the per-class scores that indicate the presence of a class instance in each of those boxes are predicted at each feature map cell [9].

Since anchor boxes constrain the location prediction, the parametrization is easier to learn, making the network more stable. Using dimension clusters along

Table 2.1: YOLO v2 Darknet-19 used for classification [33].

Type	Filters	Size/Stride	Output
Convolutional	32	3x3	224x224
Maxpool		2x2 / 2	112x112
Convolutional	64	3x3	112x112
Maxpool		2x2 / 2	56x56
Convolutional	128	3x3	56x56
Convolutional	64	1x1	56x56
Convolutional	128	3x3	56x56
Maxpool		2x2 / 2	28x28
Convolutional	256	3x3	28x28
Convolutional	128	1x1	28x28
Convolutional	256	3x3	28x28
Maxpool		2x2 / 2	14x14
Convolutional	512	3x3	14x14
Convolutional	256	1x1	14x14
Convolutional	512	3x3	14x14
Convolutional	256	1x1	14x14
Convolutional	512	3x3	14x14
Maxpool		2x2 / 2	7x7
Convolutional	1024	3x3	7x7
Convolutional	512	1x1	7x7
Convolutional	1024	3x3	7x7
Convolutional	512	1x1	7x7
Convolutional	1024	3x3	7x7
Convolutional	1000	1x1	7x7
Avgpool		Global	x1000
Softmax			

with directly predicting the bounding box center location improves YOLO by almost 5% over the version with anchor boxes [33].

Usually, objects in datasets have a typical height and width ratios. So in order to define the initial size and shape of the anchor boxes, YOLO v2 uses the k-means clustering on the dimensions of ground truth bounding boxes to get good priors for the model. Thus, when predicting bounding boxes the location and size is defined from the initial default boxes, where the definition of anchor boxes of varying sizes supports multi-scale detection.

For detection, the Darknet-19 network predicts B bounding boxes at each

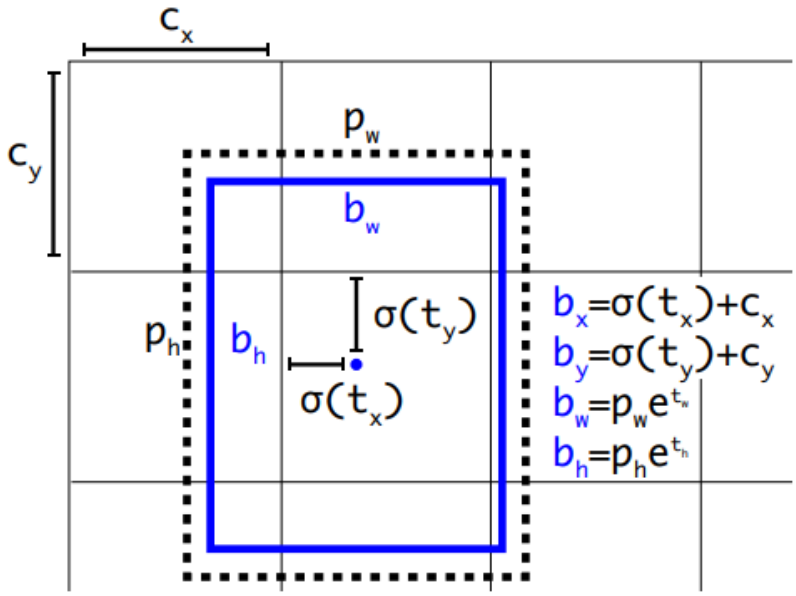


Figure 2.10: Bounding boxes with dimension priors and location prediction [33].

cell in the output feature map; This is, 5 coordinates for each bounding box, t_x, t_y, t_w, t_h, and t_o. If the cell is offset from the top left corner of the image by (c_x, c_y) and the bounding box prior has width and height p_w, p_h, then the predictions correspond to the equations 2.12 and illustrated in Figure 2.10

$$\begin{aligned} b_x &= \sigma(t_x) + c_x, \\ b_y &= \sigma(t_y) + c_y, \\ b_w &= p_w e^{t_w}, \\ b_h &= p_h e^{t_h}, \\ Pr(object) * IOU(b, object) &= \sigma(t_o). \end{aligned} \tag{2.12}$$

The structure of the Darknet-19 along with the networks design allows YOLO v2 to achieve 78.6% mAP in the VOC2007 and 73.4% in the PASCAL VOC2012 test dataset.

2.2.2 Object recognition and detection on small datasets

The goal of machine learning algorithms is to discover statistical structures within data, but building modern models such as Convolutional Neural Networks (CNNs) from scratch requires a large amounts of labeled training data [36]. When the labeled training data is limited, data synthesis and transfer learning are the commonly used approaches to build models that carry out the detection or recognition tasks achieving satisfactory performance. In addition, new research topics are standardizing such tasks under the concepts of zero-shot, one-shot, and few-shot learning; where the proposed methods are designed to specifically handle the data scarcity.

Data synthesis

Data synthesis focuses in producing artificial data either manually from scratch or using advanced data manipulation techniques to produce novel and diverse training examples. When creating new exemplars from existing instances, the available data must be enough to render novel and diverse instances to produce novel and diverse training examples [37].

As presented in the data synthesis survey in [37], plenty of research has been done aiming to improve object detection approaches using synthetic data. For instance, authors in [35; 38; 39; 40; 41] have proposed the use of synthetic data to improve detection results in fields such as robotics, showing that the results transfer to real world evaluation data.

Authors in [41] tested modern object recognition architectures such as VGG [89], Inception [42], and ResNet [91] by fine-tuning them on the ADORESet dataset [41], that contains 2500 real and 750 synthetic images for each of 30 object categories in the context of robotic manipulation; they find that a hybrid dataset achieves much better recognition quality compared to purely synthetic or purely real datasets [37].

Recently the work presented in [43] studied the question of how much real data is actually needed for object detection in comparison to synthetic data, and compared different modes of training for a number of synthetic and real datasets.

The conclusion of that work showed that fine-tuning models trained on synthetic datasets with a small amount of real data is preferable to mixed training on a hybrid dataset with the same amount of real data, and that photorealism appears to be less important than the diversity of synthetic data [37].

Another example of data synthesis can be found in [44], where the "Open Logo Detection Challenge" is proposed, along with the Context Adversarial Learning (CAL) approach. The proposed CAL takes as input initial synthetic images with logo objects to generate the corresponding context consistent synthetic images. These output images serve as additional training data for enhancing state-of-art detection model generalisation on real-world unconstrained images. In general, the data synthesis approach serves well for cases in which at least a few samples of the target categories are available but its effectiveness is limited for one-shot object recognition/detection.

Transfer learning

Transfer learning is one of the most relevant tools in machine learning, proposed to solve the insufficient training data problem. It aims to transfer the knowledge from a source domain to a target domain by relaxing the assumption that the training data and the test data must be independent and identically distributed. This leads to a great positive effect on many domains that are difficult to engage with because of insufficient training data [45]. Fine-tuning remains the method of choice for transfer learning with neural networks in which a model is pre-trained on a source domain (where data is often abundant), the output layers of the model are adapted to the target domain, and the network is fine-tuned via back-propagation. This approach was pioneered in [46] by transferring knowledge from a generative to a discriminative model, and has since been generalized with great success [47][48][49].

Traditional approaches for fine-tuning include truncating the last layer (softmax), re-train using a small learning rate, and to freeze the weights of the pre-trained first few layers. Frameworks such as Pytorch, Caffe and Tensorflow offer support to achieve these operations. This method also requires at least a few samples of the target domain categories to be available in order to perform the

fine-tuning training process, which becomes a limiting factor when dealing with one-shot detection or recognition.

Zero-shot learning

Some other machine learning based approaches have been proposed in total absence of training data. Zero-shot learning is a supervised learning approach that aims to recognize objects whose instances may not have been seen during training. Early works of zero-shot learning make use of the attributes within a two-stage approach to infer the label of an image that belong to one of the unseen classes. In the most general sense, the attributes of an input image are predicted in the first stage, and then its class label is inferred by searching the class which attains the most similar set of attributes.

In zero-shot learning, some form of side information is required to share information between classes so that the knowledge learned from seen classes is transferred to unseen classes. One popular form of side information is attributes, i.e. shared and nameable visual properties of objects. However, attributes usually require costly manual annotation. Thus, there has been a large group of studies which exploit other auxiliary information that reduces this annotation effort. The work presented in [50] took a different approach and learned class embeddings using human gaze tracks showing that human gaze is class specific [51].

Existing zero-shot learning (ZSL) models typically learn a projection function from a feature space to a semantic embedding space (e.g. attribute space). However, such a projection function is only concerned with predicting the training seen class semantic representation (e.g. attribute prediction) or classification. When applied to test data, which in the context of ZSL contains different (unseen) classes without training data, a ZSL model typically suffers from the project domain shift problem [52].

In [52] a novel solution to ZSL based on learning a Semantic AutoEncoder (SAE) is presented. Taking the encoder-decoder paradigm, an encoder aims to project a visual feature vector into the semantic space as in the existing ZSL models. The encoder projects the input data into the hidden layer with a

lower dimension and the decoder projects it back to the original feature space and aims to faithfully reconstruct the input data. A conventional autoencoder is unsupervised and the learned latent space has no explicit semantic meaning. With the proposed Semantic AutoEncoder (SAE), we assume that each data point also has a semantic representation, e.g., class label or attributes. In the object recognition task, the SAE approach achieves approximately a 14% accuracy assessed by the authors of the method in the CUB-200 dataset.

Although, state-of-art zero-shot learning approaches for object recognition have shown some progress, they ignore data available for the detection/recognition task in the context of this thesis.

Few-shot learning

More recently, a set of approaches for few-shot recognition and suitable for one-shot object recognition known as meta-learners have been proposed. The goal of meta-learning is to train a model on a variety of learning tasks, such that it can solve new learning tasks using only a small number of training samples. The work presented in [67] proposes a general and model-agnostic meta-learning algorithm. The algorithm is model-agnostic in the sense that it can be directly applied to any learning problem and model that is trained with a gradient descent procedure. The key idea behind the proposed approach is to train the models initial parameters such that the model has maximal performance on a new task after the parameters have been updated through one or more gradient steps computed with a small amount of data from that new task.

In [68] the few-shot learning problem is addressed from the key issue of overfitting. The presented approach, Prototypical Networks, is based on the idea that there exists an embedding in which points cluster around a single prototype representation for each class. The approach proposes the use of a non-linear mapping of the input into an embedding space and take a classs prototype to be the mean of its support set in the embedding space. Classification is then performed for an embedded query point by simply finding the nearest class prototype. This approach can be applied to zero-shot learning by giving a high-level description of the class rather than a small number of labeled examples.

Memory network methods for few-shot learning such as SNAIL [69] and TADAM [70] learn to store experience from seen tasks and then generalize that to unseen tasks using a meta-learner architecture consisting of interleaved temporal convolutions and causal attention layers [71]. The convolutional networks learn a common feature vector for the training instances (images) to aggregate information from past experiences. The causal attention layers learn which pieces of information to pick out from the gathered experience to generalize to new tasks. At the time this research was published it achieved the higher accuracy over the miniImageNet dataset for one-shot object recognition.

A meta-learning approach called meta-transfer learning (MTL) is proposed in [71]. The method helps deep neural nets converge faster while reducing the probability to overfit when using few labeled training data by leveraging the advantages of both transfer and meta learning. The approach consists of three phases. First, training on large-scale data to define a feature extractor. Then, a meta-transfer learning phase where the MTL model learns the parameters for the feature extractor neurons that enable fast adaptation to few-shot tasks. And finally, the typical meta-test phase is performed. At the time of this review this is the meta-learning approach for few-shot learning that achieves the highest accuracy on the miniImageNet dataset.

Following the section 1.2 this thesis focus in the particular case of a single sample available to perform detection specifically when the single sample remains unseen during the model training. The following subsection presents the literature review for the specific one-shot detection and recognition case.

One-shot learning, detection and recognition

A relatively small amount of works using machine learning has been proposed in recent years for one shot learning. The seminal work towards one-shot learning dates back to the early 2000s with the work in [53]. The authors developed a variational Bayesian framework for oneshot image classification using the premise that previously learned classes can be leveraged to help forecast future ones when very few examples are available from a given class [53] [54]. More recently, [55] approached the problem of one-shot learning from the point

of view of cognitive science, addressing one-shot learning for character recognition with a method called Hierarchical Bayesian Program Learning (HBPL). In a series of several papers, the authors modeled the process of drawing characters generatively to decompose the image into small pieces. The goal of HBPL is to determine a structural explanation for the observed pixels. However, inference under HBPL is difficult since the joint parameter space is very large, leading to an intractable integration problem.

Some researchers have considered other modalities or transfer learning approaches [55]. This recent work uses a generative Hierarchical Hidden Markov model for speech primitives combined with a Bayesian inference procedure to recognize new words by unknown speakers [56]. The work presented in [57] is one of the few published using Bayesian networks to predict attributes for Ellis Island passenger data. In [58] the one-shot learning problem in the context of path planning algorithms for robotic actuation is addressed. [65] focuses on how to borrow examples from other classes in the training set by adapting a measure of how much each category should be weighted by each training exemplar in the loss function.

The work presented in [66] explores a learning method for Siamese Neural Networks which employ a unique structure to naturally rank similarity between inputs. Once a network has been trained, it can then capitalize on powerful discriminative features to generalize the predictive power of the network not just to new data, but to entirely new classes from unknown distributions. Using a convolutional architecture, this approach presents good results in the omniglot dataset [72] improving the state-of-art in the one-shot learning context results at the time of being published.

Siamese Neural Networks

A relatively new set of architectures called Siamese convolutional networks have been later proposed to tackle to one-shot problem for tasks such as character recognition. The research presented in [66] and [73] explore the use of Siamese Neural Networks to rank similarity between a set of inputs and perform one-shot recognition using the Omniglot dataset. Once the proposed network is

trained, it can then capitalise on powerful discriminative features to generalise the predictive power of the network not just to new data, but to entirely new classes from unknown distributions.

The same approach has been applied in the context of person re-identification. In [74], a network architecture consists of two dependent convolutional networks that act independently over the two input images is presented. The networks produce an output vector for each input image, and the two output vectors are compared using a Cosine function resulting in a similarity score.

Siamese Neural Networks have also drawn attention in visual tracking given their balanced accuracy and speed [75]. Usually the approaches for tracking that involving SNNs exhaustively test all possible locations of the tracked object in the target image and choose the candidate with the maximum similarity to the past appearance of the object using the SNN output similarity metric frequently over an AlexNet backbone, as pointed in [76]. In a likely manner, the research presented in [77] proposes a human face verification multi-stage approach that aligns faces to a general 3D shape model. The authors experimented with a Siamese Neural Network where the $L1$-distance between two face features vectors is optimized to perform identity matching. The proposed multi-class network is trained to perform the face recognition task on over four thousand identities.

Siamese Neural Networks are presented in literature as a feasible approach for the problem defined in this thesis (section 1.2). SNNs remap the recognition problem as a matching problem, leveraging the use of learned features from known patterns in the training set. After training, SNNs are able to perform inferences over unseen categories without further learning/fine-tuning which poses a desired feature in the context of this thesis. The technical details of Siamese Neural Networks are explored in chapter 4 along with a performance study on the OpenLogo dataset using different embedded state-of-art CNN models.

Figure 2.11: Two sample frames from the Lasie dataset.

2.3 Experimental datasets

The following subsections introduce datasets relevant to the experimental sections of this thesis for one-shot object recognition and detection discussion, along with the metrics and libraries used for the evaluation of the assessed methods.

2.3.1 Lasie dataset

The Lasie dataset developed for the Lasie project [2] consists of 148 videos collected by the London MET police after the London Riots in 2011 as evidence for the proceeding police and legal cases. The dataset consists of a total of 1,562,516 frames, taken from different CCTV cameras in public areas around the London borough of Hackney. The videos where manually annotated for the presence of different concepts and objects in each frame aiming to develop riot detection and one-shot object detection systems. Two examples of frames from the dataset can be seen in Figure 2.11.

The Lasie dataset is used in this thesis under the "video analysis for offender identification" use case, designed to support police forces to track suspect individuals across several CCTV videos. The videos were acquired through the MET Police as part of LASIE project [2] collaboration with the MMV research group at QMUL. Appropriate data protection measures have been implemented and the videos have only been used for academic purposes.

Figure 2.12: Subset sample from the QMUL-OpenLogo dataset.

Access to the LASIE dataset is limited given its confidential nature. For this reason, the dataset was used in this thesis during the early experiments on chapter 3. Later chapters of the thesis rely on publicly available datasets.

2.3.2 QMUL-Openlogo dataset

The QMUL-OpenLogo dataset[1] is introduced in [44] as part of the "Open Logo Detection Challenge", presented as a challenging benchmark to evaluate detection methods under realistic conditions. The QMUL-OpenLogo Dataset contains approximately 27,000 images from more than 300 logo classes captured from different sources. Figure 2.12 presents a small sample set that depicts the variance between the captures and the classes in the dataset.

Given the tedious process of logo class selection, image data collection and filtering, as well as fine-grained bounding box annotation, the OpenLogo dataset re-exploits the existing logo detection datasets; i.e., the dataset contains logo images from several different existing datasets and heterogeneous sources in an attempt to reproduce realistic circumstances [44]. These datasets are: FlickrLogos-27 [78], FlickrLogos-32 [79], Logo32plus [80], BelgaLogos [81], WebLogo-2M [82], Logo-In-The-Wild [83], SportsLogo [84]. To the best of our knowledge, the

[1] `https://qmul-openlogo.github.io/`

OpenLogo dataset has not been used by any published work in the context of one-shot recognition.

The QMUL-OpenLogo benchmark has three main characteristics: (1) Highly imbalanced logo frequency distribution with some categories having as little as 10 samples and some others having several hundreds; (2) Significant logo scale variation with a range between 0.0014% 100.0% of the target image; (3) Rich scene context as shown in Figure 2.12. These factors profile the QMUL-OpenLogo dataset as a feasible alternative to approach the distinctive region or pattern detection task that motivated this research in the context of the LASIE project.

2.3.3 MiniImagenet dataset

MiniImageNet[2] is a dataset derived from the ImageNet [59] dataset and first proposed by [73]. MiniImageNet consisting of 60, 000 colour images of size 84×84 with 100 classes, each having 600 instances. This dataset is more complex than CIFAR10 [64], but fits in memory on modern machines, making it very convenient for rapid prototyping and experimentation. To construct miniImageNet 100 random classes from ImageNet are selected, the first 80 and used for training, and the last 20 for testing. Note that the last 20 class objects are never seen during training following the one-shot problem constraints. Figure 2.13 presents a set sample images from the miniImageNet dataset.

2.3.4 Pascal VOC dataset

Pascal VOC dataset is a collection of datasets for object detection [60], used for the Pascal Visual Object Classes challenge. The most commonly used combination for benchmarking traditional object detection approaches is using the VOC 2007 trainval and 2012 trainval sets for training and the VOC 2007 test set for validation. These datasets contain 10.000 and 22.500 images respectively, and each image is annotated with one or several labels, corresponding to 20 object

[2]The exact 600 images that comprise each of the 100 classes in miniImageNet can be found at `https://goo.gl/e3orz6` [73], and the images can be directly downloaded from `https://github.com/yaoyao-liu/mini-imagenet-tools`

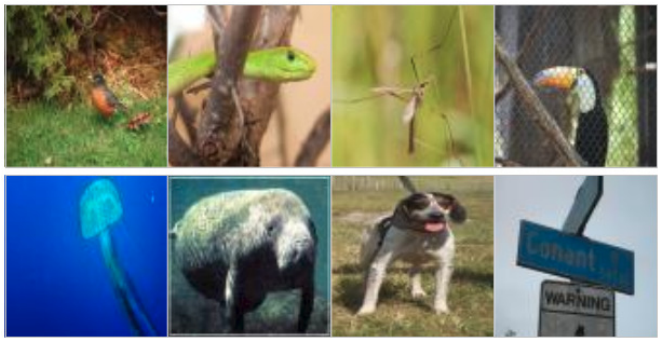


Figure 2.13: Subset sample from the miniImageNet dataset.

Figure 2.14: Subset sample from the Pascal VOC dataset.

categories. Figure 2.14 presents a of set sample images from the VOC dataset.

2.3.5 MS-COCO dataset

The COCO (Common Objects in Context) is a dataset that places the question of object recognition in the context of the broader question of scene understanding. The dataset is built from images of complex everyday scenes containing common objects in their natural context. The COCO dataset contains over a hundred thousand annotated images over 80 categories. The object categories in the data include animals, transport vehicles and furniture [61]. Figure 2.15 presents a of set sample images from the COCO dataset.

Figure 2.15: Subset sample from the COCO dataset.

2.3.6 Evaluation metrics

This section introduces the metrics used to assess the performance of the proposed approaches for one-shot object recognition and detection in this thesis. One-shot object recognition results will be presented using recall or true positive rates (TPR), False positive rates (FPR), accuracy, precision and F_1-scores. The *ROC* curve will be used to visually display the performance of the different approaches and threshold levels presented in chapters 4, and 5. Equations 2.13 to 2.17 defined the metrics used.

$$TPR = \frac{TP}{TP + FN} \tag{2.13}$$

$$FPR = \frac{FP}{FP + TN} \tag{2.14}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.15}$$

$$precision = \frac{TP}{TP + FP} \tag{2.16}$$

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{2.17}$$

where TP, TN, FP, and FN are the true positive, true negative, false positive, and false negative counts respectively; TPR is the true positive rate or recall. In particular, for the detection task the TP, TN, FP, and FN are measured following the intersection over union (IOU) definition and threshold.

In addition, the proposed one-shot object detection approach presented in

chapter 5 is measured using the mean average precision (mAP) metric. The mAP is a metric used to measure the accuracy of object detectors over all classes in a specific database and is calculated as the mean of the average precisions (AP) for each of the test classes. Equation 2.18 shows the definition of the average precision computed by interpolating the precision at each recall level in the precision vs recall curve.

$$AP = \sum_{n} (R_{n+1} - R_n) P_{interp}(R_{n+1}), \tag{2.18}$$

where R represents each observed recall level (n), and P_{interp} is defined as:

$$P_{interp}(R_{n+1}) = \max_{R:R>=R_{n+1}} P(R) \tag{2.19}$$

The AP is obtained by interpolating the precision at each recall level R, taking the maximum precision whose recall value is less than or equal to R_{n+1}. Then the mAP can be defined as the average AP over all the test classes as shown in equation 2.20.

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i, \tag{2.20}$$

where AP_i is the average precision for the ith class and N is the total number of classes being evaluated [62][63].

2.3.7 Machine learning and image processing libraries

Three main libraries were required to implement the practical compoment of this thesis: OpenCV[3], Tensorflow[4], and Pytorch[5]. OpenCV is a popular open source computer vision and machine learning software library that provides a common infrastructure for computer vision applications. The library has more than 2500 optimized algorithms. Specifically for this thesis, it was used to implement the approaches tested in chapter 3 and in general for reading, writing and operating

[3] `https://opencv.org/`
[4] `https://www.tensorflow.org/`
[5] `https://pytorch.org/`

image files.

TensorFlow is an open source library for numerical computation and large-scale machine learning developed by Google's Brain team. TensorFlow focuses on simplicity and ease of use, intuitive higher-level APIs, and flexible model building on any platform. It is probably the most popular ML framework and has wrappers for python, javascript and swift. Initial tests with knowledge transfer and data augmentation that led to the approach definition in chapters 4 and 5 were developed using Tensorflow.

Finaly, Pytorch is an open source Python-based machine learning library that provides maximum flexibility and speed. It is initially developed by Facebook artificial-intelligence research group, and Ubers Pyro software for probabilistic programming which is built on it. PyTorch redesigns and implements Torch in Python while sharing the same core C libraries for the backend code. The study presented in chapter 4 as well as the approaches in chapter 5 were developed using pytorch.

2.4 Concluding remarks

This chapter presents a state-of-art literature review including all the relevant topic to the one-shot object recognition and detection problems addressed in this thesis. In particular, the review goes one step forward in deepness regarding the Siamese Neural Networks and one-stage object detection models given their particular importance to later chapters of the thesis.

Summarizing, the relevant literature offers two main available set of approaches to address the one-shot object recognition and detection problems. On one hand, image processing based approaches; popular before Convolutional Neural Networks came to existence, tend to work well on very constraint environments but usually fall short on realistic scenarios due the inherent variability of the patterns to match along the test datasets. These approaches are explored further in chapter 3. On the other hand, approaches base on the use of Convolutional Neural Networks have shown high performance in several application fields. Such approaches are usually constraint by the amount of data they re-

quired to properly train the models.

Siamese Neural Networks appear in recent literature as a feasible option to engage with the one-shot object recognition problem. This approach is designed to fir the one-shot object recognition constraints, but most of the success use cases found in the literature are related to basic computer vision problems, such as character recognition. Chapter 4 explores the potential of SNNs to face the one-shot object recognition problem testing over a complex and realistic dataset. Finally, the theory presented in this review is used to develop a feasible alternative for one-shot object recognition and detection in chapter 5.

Chapter 3

Image Processing Approach for One-shot Detection

Previous to Siamese Neural Networks, approaches developed to tackle the one-shot problem were mostly based on image processing techniques. Although Siamese neural networks (SNN) were first introduced in mid 90s by LeCun [85] they only started gaining attention in the one-shot recognition problem context after Koch's paper [66] in 2015. As introduced in chapter 2, image processing approaches either define features or keypoints over regions within a query and target images to be compared in later stages of the detection process. In consequence, this chapter presents the experiments performed in the early stages of this research around image processing approaches to engage in the one-shot recognition problem.

The study presented in this chapter focused on the exploration of image processing approaches in the context of video analysis for offender identification within the Lasie project; where the main goal is to automatically detect salient regions in images or video frames from surveillance data to facilitate identification of a suspects. Three promising approaches are selected for the study. First, the SURF [12] approach based on keypoint detection and matching; second, the locally adaptive regression kernels [21] based on gradient estimation initially proposed for image denoising, interpolation, and deblurring; Third, a

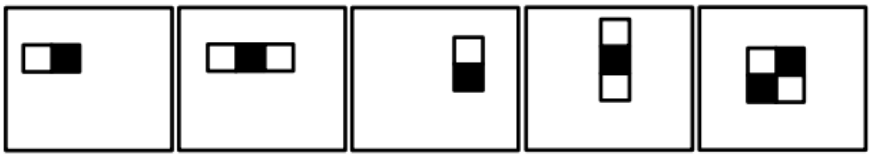


Figure 3.1: Haar-like features proposed in [86]

proposed modification of the viola-jones [86] approach using sliding windows is introduced. The approaches are tested in a subsample of the CCTV videos of the Lasie project dataset.

3.1 Viola-Jones algorithm

This section explains in detail the Viola-Jones algorithm from [86] and its application for one-shot object detection. First, an explanation of the Haar-like features used by the algorithm and how they are computed using integral image is presented. Then, the learning functions proposed by the approach and how they are assembled into a attentional cascade classifier is explained. Figure 3.1 shows the five types of features first proposed in the Viola-Jones algorithm. There are 162.336 possible Haar-like features that would fit in a 24×24 support window.

Each feature value computed from the difference between the sum of gray level intensities on the white and gray regions. More precisely, the feature value f for a feature located in the coordinates x, y with $w \times h$ size in and image I is given by equation 3.1:

$$f = \sum_{i=x}^{x+w} \sum_{j=y}^{y+h} I(i,j)_{inwhiteregion} - \sum_{i=x}^{x+w} \sum_{j=y}^{y+h} I(i,j)_{inblackregion}. \tag{3.1}$$

In order to reduce the computational cost of calculating the feature values the concept of integral image is introduced. The pixel values $II(x, y)$ of an integral image II computer from an image I contains the sum of all the gray intensity values above and to the left inclusive of the image I as shown in Figure 3.2 and

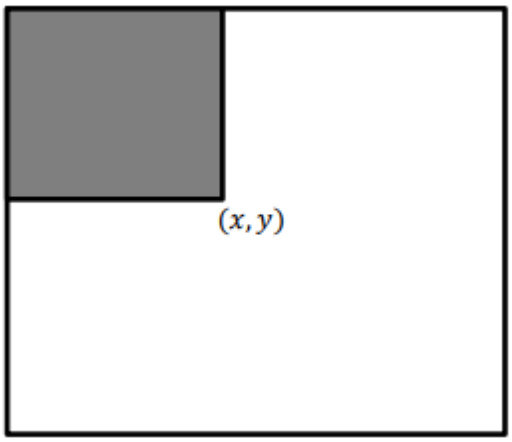


Figure 3.2: The value of the integral image at a point (x, y) is equals to the sum of all the pixels above and to the left

equation 3.2.

$$II(x,y) = \sum_{x'=0}^{x} \sum_{y'=0}^{y} I(x', y'). \tag{3.2}$$

By using integral images, the sum of the pixels inside a rectangular region can be reduced to a basic operation between 4 pixel values as shown in Figure 3.3 and equation 3.3.

$$gray_{area} = D + A - (B + C). \tag{3.3}$$

Each one of the possible Haar-like features $f(x)$ in a 24×24 window can be mapped to a weak classifier $h(x, f, p, \theta)$ as shown in equation 3.4:

$$h(x,f,p,\theta) = \begin{cases} 1, & pf(x) < p\theta \\ 0, & otherwise \end{cases}. \tag{3.4}$$

The goal of the Viola-Jones algorithm is to create sub-sets of weak clas-

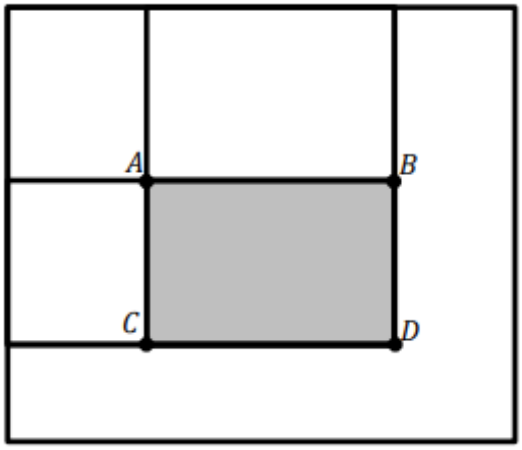


Figure 3.3: Area of the gray rectangle to be computed using equation 3.3

sifiers that best discriminate the object of interest. To achieve this goal, the adaboost algorithm is used as a feature selection process. Algorithm 1 presents the boosting algorithm to build a strong classifier $C(x)$ using the best T weak classifiers.

Algorithm 1: Viola-Jones boosting algorithm

Data: Given example images $(x_1, y_1), ..., (x_n, y_n)$, where $y_1 = 0$ or 1 for negative and positive examples respectively.

Result: strong classifier $C(x)$

Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negative and positives samples respectively;

for $t \leftarrow 1$ **to** T **do**

- Normalize weights $w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$;
- Select the best weak classifier with respect to the weighted error: $\epsilon_t = min_{f,p,\theta} \sum_i w_i |h(x_i, f, p, \theta) - y_i|$;
- Define $h_t(x) = h(x_i, f_t, p_{t\theta t})$, where $f_t, p_{t\theta t}$ minimize ϵ_t;
- Update wights: $w_{t+1,i} = w_{t+1,i} \beta_t^{1-e_i}$, where $e_i = 0$ if the sample x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$;

end

The final strong classifier is: $C(x) = \begin{cases} 1, & \sum_{t=1}^{T} \alpha_t h_t(x) > \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0, & otherwise \end{cases}$

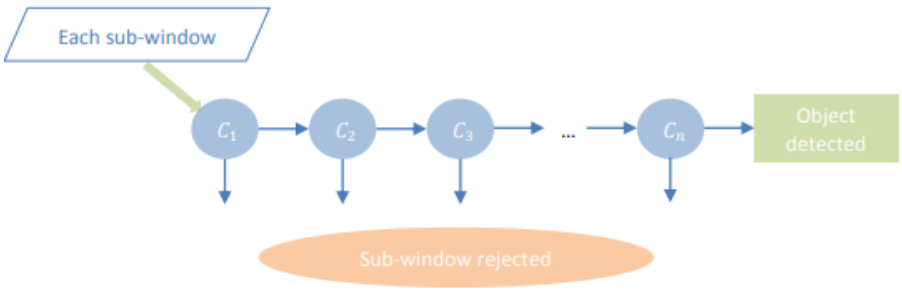


Figure 3.4: Attentional cascade classifier architecture.

The classifier used for detection on the Viola-Jones approach is called Attentional cascade classifier. This classifier is composed of stages of classification, where, each stage is composed of a strong classifier which in turn is composed of several weak classifiers. A specific sub-window has to pass through all the stages of classification in order to be accepted as a positive detection by this approach, as shown in Figure 3.4.

Algorithm 2 shows the algorithm used to train the classifier and determine how many stages and weak classifiers each strong classifier will have.

In [86] the approach is tested for a face detection framework achieving a detection rate of 77.8% with 5 false positives in a MIT dataset containing 23 images with 149 faces. This detection rate is slightly lower than the state-of-art approaches at the time of the publication, but the detection speed is a key contribution of the work, being 15 times faster than any previous approach in face detection.

The Viola-Jones algorithm is available in the opencv library and can be used for one-shot detection by training the model applying multiple geometrical transformations to the query image.

Algorithm 2: Attentional cascade training algorithm using adaboost.

Data: The maximum acceptable false positive rate per layer f. The minimum acceptable detection rate per layer d. P: Set of positive samples. N: Set of negative samples.

$i = 0$;
while $F_i > F_{target}$ **do**
  $i = i + 1$;
  $n_i = 0$;
  $F_i = F_{i-1}$;
  while $F_i > f \times F_{i-1}$ **do**
    $n_i = n_{i+1}$;
    - Use P and N to train a classifier with n_i features using Adaboost;
    - Evaluate current cascade classifier on validation set to determine F_i and D_i;
    - Decrease threshold for the ith classifier until the current cascade classifier has a detection rate of at least $d \times D_i - 1$(this also affects F_i);
  end
  $N = 0$;
  - If $F_i > F_{target}$ then evaluate the current cascade detector on the negative validation set and put any false detections on N;
end

3.1.1 Proposed Viola-Jones modification for one-shot detection

This section describes an early proposed approach for one-shot DROP detection. This approach is inspired by the Viola-Jones attentional cascade classifier. The main differences with the Viola-Jones algorithm are, firstly, introducing the assumption of no training data available, and a single query image being to find the required DROP; secondly, the inclusion of color information in the DROP search process.

The proposed approach involves two stages: description and search. First, the target DROP is described using color and Haar-like features in order to define the conditions to be matched in the search stage. Figure 3.5 shows an overview of the class (DROP) definition process. A set of Haar-like features and color measurements are obtained as the result of this stage.

Subsequently, a sliding-window based search is performed on the target images using the attributes obtained in the definition stage. The search space for

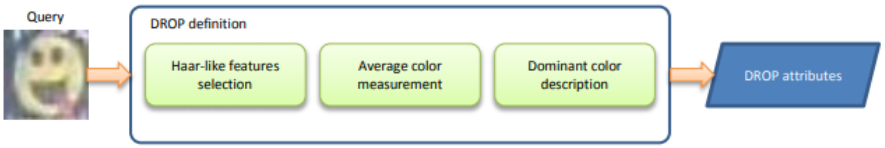


Figure 3.5: DROP Features extraction process.

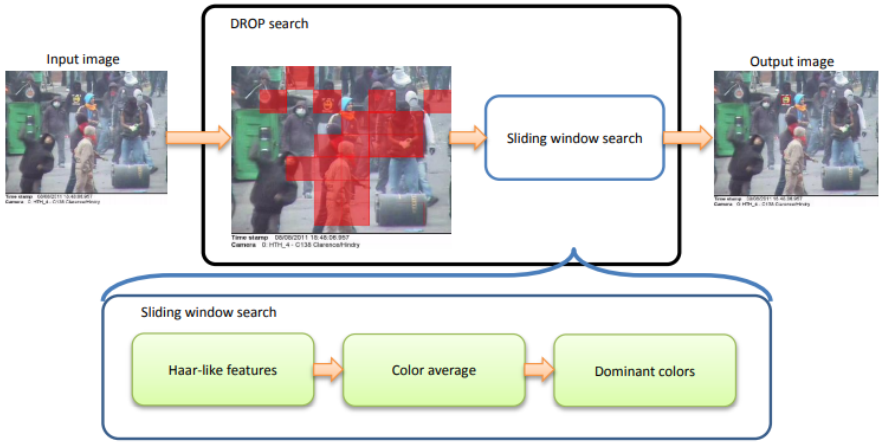


Figure 3.6: DROP Features extraction process.

the sliding window in the image is initially reduced by defining an attentional grid using a dominant color matching approach. Then, for each possible sliding-window fitting in the attentional grid, a set of conditions is assessed: Haar-like features (intensity), average color, dominant colors. These conditions are used to compare the current sliding window against the query DROP. If every condition is satisfied for a specific sliding-window, the region covered by the window is marked as a positive detection. Figure 3.6 shows the steps involved in the search process.

DROP description: In the proposed approach Haar-like features, average color and dominant color descriptor (DCD) measurements are used to describe a particular DROP. This work assumes that there is no training data available. Thus, the Haar-like features are selected using a score function that assigns high values to features representing noticeable changes in gray intensity level and regions where low or no changes are present. Equation 3.5 presents the score function used for Haar-like feature selection:

$$score_i = max(\alpha_1 v_i + \alpha_2 b_i, 127 - \alpha_1 v_i), \tag{3.5}$$

where v_i is the normalized value obtained by computing the Haar-like feature i on the query DROP. b_i is the normalized value obtained by computing the same Haar-like feature over the image borders (Sobel). α_1, α_2 are input parameters to control the relevance each term in the selection. The 127 coefficient is equivalent to the maximum normalized value of any Haar-like feature. The first n Haar-like features with higher scores are selected, with n being a user defined parameter. The first term in the maximum function assigns high scores to regions where noticeable intensity changes are present. Likewise, the second term assigns high scores to features where low intensity changes are present. In this way, the structure of the pattern is represented via a set of Haar-like features.

Finally, the selected Haar-like features are split into groups in an exponential manner in order to reduce the computational time required during the search stage. In this way, the first groups will contain few of the most relevant features and the last group will contain a greater amount of less relevant features in terms of the score function as the stated hypothesis in [86]. A threshold defined as an input parameter is used proportionally for each group in the search stage to detect the DROP.

The average color and dominant color descriptor (DCD) are also used to define the DROP. The average of the hue component of the query DROP is measured in the HSV color space as presented in equation 3.6.

$$\bar{h} = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} h_{i,j}, \tag{3.6}$$

where $M \times N$ is the query DROP size and $h_{i,j}$ is the hue value of the pixel in the position i, j in the sliding window. This average will be compared with possible matches in the detection stage using a defined threshold range by the user as an input parameter.

The DCD provides a compact description of the representative colors in an image or image region. The MPEG-7 DCD is composed by the space coordinates that represent color, the percentage of presence of each color in the assessed region, the color variance, and the color homogeneity in the image as shown in equation 3.7.

$$F = \{\{c_i, p_i, v_i, \}, s\}, (i = 1, 2, ..., N), \tag{3.7}$$

here, c_i is the color coordinates in the HSV space, $p_i$is the percentage of presence of each dominant color in the image, v_i is an optional value that describes the variation of the color values of the pixels in a cluster around the corresponding representative color, and s represents the overall spatial homogeneity of the dominant colors in the image. The number of colors used to describe the query DROP depends directly on the DROP itself, and thus, is defined via an input parameter. This work uses the similarity metric presented in [86] as shown in equation 3.8 which computes the called Mutual Color Ratio (MCR).

$$SIM(I_1, I_2) = \left(\sum_{i=1}^{M} \sum_{j=1}^{N} a_{i,j} S_{i,j} \right), \tag{3.8}$$

where $S_{i,j}$ refers to the similarity between color percentages as presented in equation 3.9; $a_{i,j}$ is the color similarity between two colors c_i and c_j as shown in equation 3.10; and M, N are the amount of dominant colors in the compared images I_1, I_2. The MCR is computed as shown in algorithm 3.

$$S_{i,j} = [1 - |p_i(i) - p_j(j)|] \times min(p_i(i) p_j(j)), \tag{3.9}$$

$$a_{i,j} = \begin{cases} 1 - \frac{d_{i,j}}{d_{max}}, & \text{if } d_{i,j} \leq Th \\ 0, & \text{if } d_{i,j} > Th. \end{cases} \tag{3.10}$$

In equation 3.10, $d_{i,j}$ is the Euclidean distance between c_i and c_j representing the maximum distance whereby two colors are considered similar; and $d_{max} = \alpha Th$, with Th being an user defined threshold.

Algorithm 3: MCR algorithm.

Initialize $MC = 0$;

- Find mutual colors by pass on all the dominant colors (DCs) of the two images;

for *DCs in input_images* **do**

  $\forall \mathrm{dc}_i \in \mathrm{DC}_1$, $i = 0, ..., N-1$;

  $\forall \mathrm{dc}_j \in \mathrm{DC}_2$, $j = 0, ..., M-1$;

  - {Count similar colors between the two images}

  if $distance(dc_i, dc_j) < Th_d$ **then**

    $MC++$;

  else

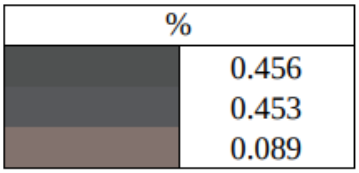
end

- {Compute the mutual radio color between two images}

$MCR = \frac{MC}{max(M,N)}$

Following the description stage, the search step is introduced as follows: Given a query DROP description, an intensive search is performed on the input media. Initially, the search space for the sliding-window approach is reduced by measuring dominant colors on cells of a defined grid in the input image. The cells are marked for search in case that any of its dominant colors are similar to the dominant colors of the query pattern. Then, a sliding-window is iterated through the search space and three main conditions are checked: Color average, Haar-like features, and DCD.

(a) Query DROP



(b) Dominant colors

(c) Target image

(d) Reduced search-space: cells marked in red

Figure 3.7: Search space reduction

Attentional grid definition: The color attentional grid is a first step in the detection stage, that aims to reduce the search space of the proposed sliding window approach. In this step, the input image is divided into a $M \times N$ grid. For each cell on the grid a number of dominant colors is computed according to a user input parameter. The dominant colors in the cell are compared to the dominant colors computed for the query DROP. A cell is marked for search if a considerable region in the cell contains dominant colors similar to the query DROP dominant colors. The similarity between dominant colors is measured using a weighted Euclidean distance in the HSV color space. Figure 3.7, shows some results of the search space reduction using this approach.

The color attentional grid discards a considerable amount of regions of the image where there is no need for searching. Therefore, it reduces the computational time of the sliding window stage. This process can also be performed offline and stored as media metadata.

Sliding window: Once the attentional grid is defined a sliding-window will iterate all the possible coordinates and sizes fitting inside the attentional grid to check the presence of the DROP using the attributes generated in the DROP

definition stage. The following paragraphs describe the conditions assessed for each iteration of the sliding-window.

Haar-like features: The Haar-like feature groups selected in the DROP definition are computed for each window. The assessment of the feature groups follow the attentional cascade architecture presented in [86] as shown in Figure 3.1. Again, the first groups contain fewer but relevant features and the latter groups contain more but less relevant features in terms of the assigned score. The accumulated difference between the features values from the DROP definition stage and the current window is computed for each group. Finally, the computed accumulated difference is used to define the similarity between the current window and the query DROP by thresholding the feature values of the selected groups with a user input parameter as shown in equations 3.11 and 3.12.

$$AD_i = \sum_{j=1}^{m} |query_{v_j} - sample_{v_j}|, \tag{3.11}$$

$$AD_i \leq T_i, \tag{3.12}$$

where AD_i is the accumulated difference for the ith features group; m is the number of features in the group i; $query_{v_j}$ and $sample_{v_j}$ are the values of the feature assessed on the query DROP image and current sliding-window respectively. Finally, T_i is a threshold defined from a user input parameter (T) for each feature group proportionally to the number of features that it contains.

Color average: The average of the Hue component in the HSV color space is computed for the region and compared to the average hue on the query pattern. A threshold that defines a range to consider a pair of colors to be similar is defined as a user input parameter. If the Haar-like features and color average condition are satisfied the dominant color condition is then computed.

Dominant color descriptor: Using DCDs a final check on color for the query DROP is performed. In this case the image region defined by the current sliding-window is clustered into dominant colors. A DCD matching distance derived from color similarity is computed over the window in order to check the presence and percentages of the dominant colors in the query DROP as shown in equation 3.13. The purpose of this condition is to reject the false positives accepted by the two previous conditions.

$$d = 1 - SIM(I_1, I_2). \tag{3.13}$$

3.2 Experiments

This section presents a performance assessment of the image processing based approaches for one-shot detection presented previously. The tests are carried out on a subset of the Lasie dataset following the video analysis for offender identification use case.

The SURF and modified Viola-Jones approaches are coded in c++ using the opencv library. The LARKs code is published by the authors[1], programmed using matlab and available for public download.

In order to achieve the one shot detection assessment, an arbitrary DROP query is selected from the dataset and the search process is performed over a 35 seconds sample video where the DROP is present. Figure 3.8 shows a query image used for the search along with a sample frame of the video where the target DROP is present.

Table 3.1 shows the search results in terms of true positive rates (TPR), false positive rates (FPR), Precision and Accuracy according to equations 2.13 to 2.16 for the SURF, LARKs, and the modified Viola-Jones approaches presented in this chapter.

Here, the LARKs approach achieves a higher performance with a 0.952 ac-

[1] `https://users.soe.ucsc.edu/~milanfar/research/rokaf/.html/ObjectDetectionFromSingleImage.html`

(a)

(b)

Figure 3.8: (a) Query DROP used for experiments, (b) Target frame sample.

Table 3.1: Search results for the three image prosecing based approaches described in this section

Approach	Modified VJ	SURF	LARKs
TPR	0.414	0.648	0.776
FPR	0.011	0	0
Precision	0.906	1	1
Accuracy	0.867	0.925	0.952

curacy compared to 0.925 and 0.867 of SURF and the modified viola approach, respectively. This improved performance comes at expenses of computational time. Whilst the SURF detection process takes around five hundred milliseconds to search one frame, the LARKS implementation takes around two seconds to perform the task. Following [21] the LARKs approach may take over to 10 seconds to compute the features of a 640x480 image particularly when accounting for size and rotation variations. This is mainly because the LARKs search process is performed on overlapped patches in a similar way to sliding windows approaches.

In this particular experiment the high accuracy can be explained by the size of the testing set, where the input parameters for each algorithm can be tune to maximize its performance. When image processing approaches are applied under general one-shot object detection scenarios the performance drops

considerably.

Such is the case of the SURF approach evaluated on the QMUL-OpenLogo dataset, resulting in a 9.80% mAP (see section 5.4.2 for experimental setting details). Whilst keeping a competitive execution time, the mAP results are noticeably low when compared to modern approaches even for one-shot object recognition. In addition to low performance, the presented kernel based approaches (LARKS, VJ) become impractical for realistic scenarios due the re-training requirements and slow performance.

3.3 Concluding remarks

This chapter presents an assessment of three different techniques for one-shot generic object detection based on image processing. The theoretical details of each technique show their feasibility to meet the one-shot detection requirements by using a single sample image of the pattern to search. Fundamentally, all the presented methods perform a two stage process in which, first, a feature extraction mechanism takes place either based on box filters (SURF and Viola-Jones) or gradient estimation (LARKs) to define both the query and target images, and secondly, a comparison of the extracted features enables the search and matching in the input images.

The presented methods rely on an initial feature extraction process over a single image to define the descriptors that characterize the input query patterns to be searched. This implies that an appropriate comparison with target regions highly depend on how complex the target changes are between the different video frames or target images. Hence, the kernel based methods are more likely to be affected by perspective changes and occlusions present in the target images. Whereas, keypoint based methods performance can be impaired by the complexity of the background in the target image due to keypoint mismatches, creating room for false positive detections and poorly defined bounding boxes.

Such factors are common in realistic datasets for object detection like the Lasie and QMUL-OpenLogo datasets, where complex images in terms of content are captured from different sources. As presented in this chapter, when the

subset of target images are very homogeneous image processing approaches may be tuned to achieve good results. However, when the task at hand includes a large set of complex queries and target images such approaches fall behind in terms of time and performance.

This chapter results demonstrate the limitations of image processing approaches for object detection handling perspective, structural, and illumination variations expected to be present in realistic scenarios. Thus, highlighting the relevance of machine learning as a feasible option to tackle the image processing limitations via modeling such variations using training data. In consequence, the following chapters of this thesis explore machine learning based solutions to tackle object recognition and detection under the training data restrictions that characterize the one-shot recognition and detection problems.

Chapter 4

Siamese Neural Networks for One-shot Recognition

Traditional neuronal network based approaches for object recognition use a single data pipeline from input to output in order to assign a predefined category to an image. Such approach is appropriate to tackle specialized tasks like classic image recognition, where an input image is associated with a particular class from a fixed and finite set of categories. The main disadvantage of the classic approach is the amount of data required to build recognition models that perform adequately. This feature is very restrictive for applications where the data compilation has a high cost or tasks such as one-shot object recognition, where the data is not available. In addition, adapting the trained models to recognize new classes is not a trivial task.

However, the premise of Siamese Neural Networks (SNNs) introduces the use of an additional input to generate adaptive outputs; in other words, Siamese networks allow to perform category matching between a pair of input images by computing a similarity metric. This property makes the Siamese networks approach desirable for the one-shot recognition problem as defined in section 1.2. Following [87] the Siamese Neural Networks can be divided into two types of architectures: pairwise models and triplet models.

The architecture of the pairwise models studied in this research requires two

input images in order to perform matching inferences. The two input images are forwarded through the twin sister networks connected at the top of the architecture, where a similarity metric is computed between the resulting feature vectors.

In contrast, the triplet models add a third input image to the architecture. In addition to the query and target input images used for matching, a third image depicts a non-matching category is selected arbitrarily. The objective is to minimize the distance between the matching target and query patterns and maximize the distance between the target and the third non-matching patterns. Although, triplet networks are commonly used to approach the one-shot problem in some application fields such as person re-identification, they require additional information per query in order to perform the recognition task. Additionally, there is not enough evidence triplet models outperforms the classic pairwise SNN architecture[87]. For this reason, the following sections and chapters in this thesis will refer to the pair wise models simply as Siamese Neural Networks.

In the relevant literature, one-shot recognition is commonly approached using SNNs embedding an AlexNet [75][76][94] architecture as the backbone of the model. Although, it is unclear whether the complexity of a shallow model like AlexNet embedded in the SNN architecture is enough to approach imagery closely related to real world applications. For this reason, this chapter presents an study of the Siamese Neural Network approach for one-shot recognition assessing several embedded convolutional Neural Network architectures, loss functions and similarity metrics.

The following sections explain the SNN architecture approach to the one-shot recognition problem along with the methodology used to evaluate different embedded models into the architecture.

4.1 Siamese Neural Networks

Siamese neural nets were first introduced in the early 1990s by Bromley and LeCun to solve signature verification as an image matching problem [85], but only after [66] started trending as a solution for the one-shot recognition prob-

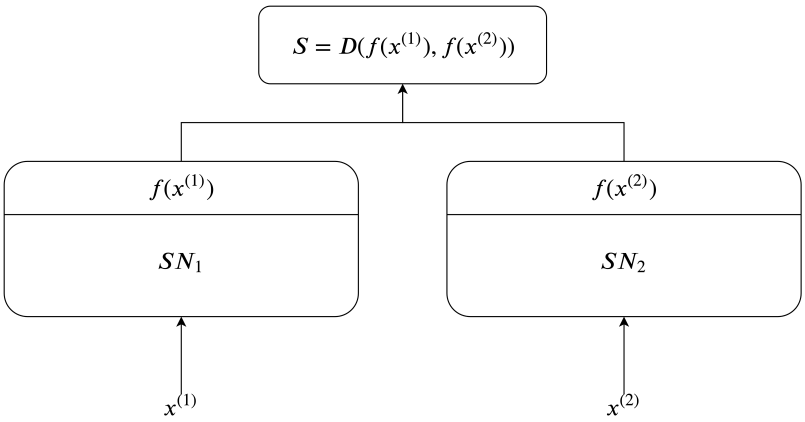


Figure 4.1: Siamese Neural Network architecture.

lem. This trend is particularly evident in application fields such as person re-identification and object tracking.

Siamese Neural Networks are a type of neural network composed of a pair of branches with tied parameters. Each branch takes an input image and generates a corresponding feature vector that are later compared using an cost function at the top of the architecture. The cost function computes a distance metric between the resulting highest-level feature representations of the two branches. Let $x^{(1)}$, $x^{(2)}$ be the input images processed through the layer sequence of each one of the architecture branches or sister networks SN_1, SN_2. The sister networks produce a "encoded" feature vector $f(x^{(1)})$ and $f(x^{(2)})$, respectively for each input image. The sister networks SN_1, SN_2 do not have any explicit restrictions over their architecture, so a CNN can be defined according to specific use cases, as shown in the experimental section of this chapter.

Architecture

Ideally, if the convolutional networks SN_1 and SN_2 encode adequately the input images $x^{(1)}$ and $x^{(2)}$, the distance between the output feature vectors $D(f(x^{(1)}), f(x^{(2)}))$ will be large for dissimilar inputs and small for similar inputs. Furthermore, the shared weights constraint on the twin CNN's guarantees that two extremely similar images could not possibly be mapped by their respective networks to very different locations in feature space given that each

network computes the same function [88]. Figure 4.1 illustrates the proposed Siamese Neural Network architecture following the discussed formalization.

Loss function

Although several loss proposals to train Siamese Neural Networks are available in the literature such as hinge or cosine losses, this work will focus on the more generally used approaches for assessment purposes. Commonly, Siamese networks are trained using a distance-based loss function called contrastive loss [66] [88]. The main goal of the contrastive loss function is to construct a Euclidean space where the outputs of the sister networks $f(x(1))$, $f(x(2))$ are close to each other for matching pairs while simultaneously negative pairs are pushed away [88]. Following [88], this research uses the constrastive loss function defined as:

$$\begin{aligned} \ell_{cl}(x_i^{(1)}, x_i^{(2)}) = \sum_{i=1}^{N} (1 - y_i)\frac{1}{2}(D^i_{l2norm})^2 + (y_i) \\ + \frac{1}{2}\{max(0, m - D^i_{l2norm})^2\}, \end{aligned} \tag{4.1}$$

where y_i equals to 1 if the input pair $(x_i^{(1)}, x_i^{(2)})$ contains matching patterns or equals 0 otherwise; m is a margin preventing dissimilar pairs to contribute to the loss; and D^i_{L2norm} is the $L2$-norm between the output feature vectors of the sister networks $f(x_i^{(1)})$ and $f(x_i^{(2)})$.

In addition to the contrastive loss function this work assesses the binary cross entropy (BCE) loss function for SNN training as defined in equation 4.2. The loss function is defined to comply with the SNN architecture previously defined and the specified one-shot constraints, where, the model is trained to output a similarity measure that is small for matches and big for dissimilar mismatches.

$$\ell(x_i^{(1)}, x_i^{(2)}) = -\frac{1}{N}\sum_{i=1}^{N} y_i(x_i^{(1)}, x_i^{(2)}) \cdot log(p(x_i^{(1)}, x_i^{(2)})) \\ +(1 - y_i(x_i^{(1)}, x_i^{(2)}) \cdot log(1 - p(x_i^{(1)}, x_i^{(2)})), \tag{4.2}$$

where, y_i is the matching label between the inputs $x^{(1)_i}, x^{(2)_i}$, which equals to 1 for matching input pairs and 0 otherwise. $p(x^{(1)}, x^{(2)})$ is the predicted label and N is the batch size.

4.2 Methodology

The study presented in this chapter aims to evaluate the performance of a set of convolutional networks embedded into the Siamese Neural Network architecture using the QMUL Open-Logo dataset. An assessment of the introduced Siamese model is carried out using a set of CNNs as the backbone for the architecture; that is, a group of state-of-art models is embedded as SN_1 and SN_2 into the SNN architecture to perform the one-shot recognition task. The convolutional Neural Networks used for this evaluation are: AlexNet [23], VGG [89], DenseNet [90], ResNet [91], and the architecture proposed in [66] for one-shot character recognition.

For the first set of experiments the convolutional networks embedded as SN_1 and SN_2 in the Siamese architecture are pretrained using the ImageNet dataset. For each CNN the Siamese network is fine-tuned on the QMUL Open-Logo dataset using the BCE loss function. The fine-tuning strategy takes advantage of the learned features from the pre-trained models to carry out the one-shot logo recognition task over the challenging dataset. Additionally, the output similarity metric will be computed using a fully connected layer of a single neuron over the absolute difference of the feature vectors $f(x^{(1)})$ and $f(x^{(2)})$ in a similar manner to [66].

After evaluating the performance of the Siamese network with each one of the proposed embedded models, the best performing model is used to carry out additional experiments that may exhibit specific points for performance im-

provements. In particular, these aspects include the loss function, the similarity layer at the top of the SNN, and the learning approach.

First, the mainstream contrastive loss function is compared against the binary cross entropy loss function. Then, the performance of the Euclidean distance that defines the similarity metric D is also compared to the output given by a linear layer of a single output neuron denoting D and also two output neurons defining match and mismatch categories; therefore, avoiding the use of thresholds. Finally, the learning approach is assessed by comparing the fine-tuned model against the same model trained from scratch and using pre-learned features only.

Fine-tuning: Following [45], deep transfer learning may be classified into four categories: Instances-based, Mapping-based, Network-based, and Adversarial-based. Specifically the for one-shot object recognition problem, given that there are no prior samples of the target object and ideally no time should be expended retraining the model for each new query pattern; this study uses Network-based transfer learning. In this case, the network pretrained in the source domain is reused partially.

According to [45], Given a target learning task T_t based on a target domain D_t, that can be supported from a source domain D_s for the source learning task T_s. Transfer learning aims to improve the performance of predictive function $f_T(\cdot)$ for learning task T_t by discovering and transfering latent knowledge from D_s and T_s, where $D_s \neq D_t$ and/or $T_s \neq T_t$. This process is depicted in Figure 4.2.

In this particular case, the source and targets tasks T_s, T_t are image recognition, where T_t is specifically focused on one-shot recognition. The source domain D_s is generic object recognition over the ImageNet dataset [92], where a thousand categories are represented and learned by the models to be fine-tuned; and the target domain D_t is one-shot logo recognition.

Learning: In accordance with the goals of this thesis, the introduced Siamese Neural Network architecture is lined up to perform one-shot recognition over unseen classes. Therefore, the experimental dataset is split into non overlapping training and testing categories guaranteeing that the SNN has no knowledge of

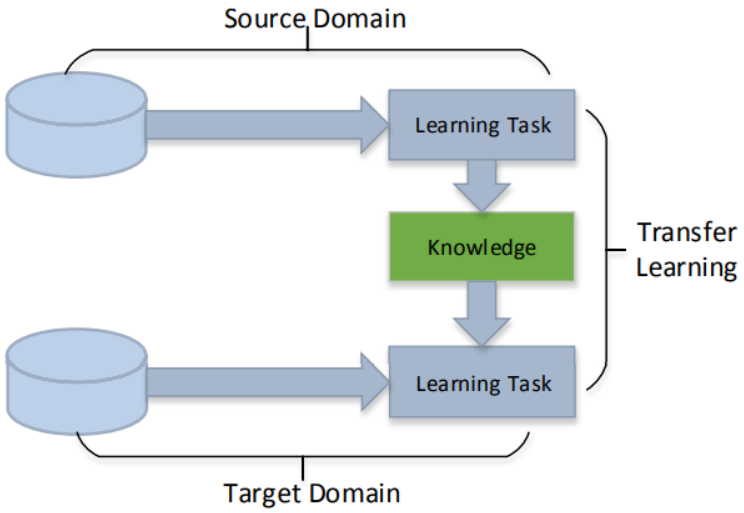


Figure 4.2: transfer learning process overview [45].

any of the test query inputs and complies with the one-shot constraints.

The QMUL Open-Logo dataset [44] is divided 60/40 approximately, between training and testing categories for the experimental section; this is, 211 categories for fine tuning and 125 categories for testing. As explained in section 2.3.2 the amount of instances per classes in the dataset is unbalanced, with some of the logos having as little data as 2 instances. To address this issue the data loading process is programmed to load the different categories with matching and mismatching pairs with equal probability.

The dataset is originally assembled for the object detection task. In order to adapt it for the one-shot object recognition the dataset is first cropped using the bounding boxes from the ground truth annotations, so that only the region of interest in the image is used to perform the recognition task.

From the pre-trained models embedded in the Siamese architecture the fine-tuning process takes place. The fine-tuning process iterates 200 epochs for each embedded architecture with a learning rate of 0.005. Each training batch contains 64 pairs of images randomly selected from the training set with a 50/50 split between matches and mismatches. For testing purposes, approximately 20,000 input pairs are formed from the 125 testing categories available, again with a 50/50 split between matches and mismatches.

Implementation details: The proposed framework is coded based on the pytorch library using the `torchvision.models` package for the proposed embedded architectures listed as follows: `alexnet` [23], `vgg16` [89], `densenet161` [90], `resnet18` [91]. Additionally, the architecture proposed in [66] is implemented from scratch and embedded in the framework for comparison purposes. For each pre-trained architecture, all layer weights are frozen and only the last layer is reset for fine-tuning as motioned earlier in this section. The full code can be found at `https://github.com/cjvargasc/oneshot_siamese/`.

Once the pretrained models are downloaded, the fine-tuning process takes place for $epochs = 200$, and $batch\_size = 64$ for each architecture. The fine-tuning process took from 3 to 5 hours with the current training/fine-tuning protocol, excepting the `vgg16` model, that took around 10 hours of training to complete the 200 epochs on a GeForce GTX 1080.

Recognition results will be presented using true positive rates, false positive rates, accuracy, precision and $F1$-scores, following equations 2.13 to 2.17. The *ROC* curve will be used to visually display the performance of the different embedded CNNs and threshold levels applied over the similarity metric D. Given that different thresholds lead to differences in the performance of the binary classifier, a set of 20 thresholds is applied on the similarity metric within the range of responses returned by the Siamese Neural Network.

4.3 Experiments

This section presents the results of the introduced approach for one-shot logo recognition. Firstly, a performance comparison of the Siamese architectures with the different embedded models is carried out. Secondly, the embedded CNN model with the best performance scores is used to compare different learning approaches; this is, training the whole Siamese model from scratch or by using the pre-learned features only without further fine-tuning. Finally, the same approach is used in order to assess different similarity metric layer strategies.

Figure 5.5 shows the *ROC* curve resulting from testing the selected embedded models in the proposed architecture for one-shot logo recognition. The

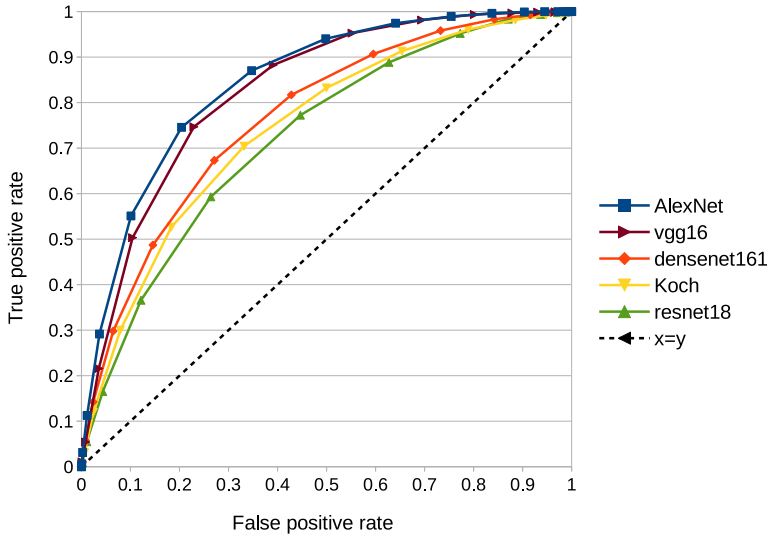


Figure 4.3: *ROC* curve showing the performance of the proposed approach on the QMUL-OpenLogo dataset.

ROC curve is obtained by thresholding the range of responses of the similarity metric layer output obtained using each embedded CNN. All the models are fine-tuned using the BCE loss function, with a single neuron output for the last layer as in [66].

At each threshold level the TP, TN, FP, and FN are counted to compute the true positive rate (TPR), false positive rate (FPR) rate, accuracy, precision, F_1-scores, and the area under the *ROC* curve (AUC). Table 4.1 shows the detailed results for each one of the embedded models in the proposed framework. The scores presented in Table 4.1 are reported for the threshold level that achieves the highest accuracy.

Table 4.1: Performance metrics for the proposed approach using different embedded CNNs.

Embedded CNN	TPR	FPR	acc	Pr	F1	AUC
$AlexNet$	0.74	0.20	0.77	0.78	0.76	0.84
vgg	0.74	0.22	0.75	0.76	0.75	0.82
$Koch$	0.70	0.33	0.68	0.67	0.69	0.74
$Resnet$	0.59	0.26	0.66	0.69	0.63	0.72
$denseNet$	0.67	0.27	0.70	0.71	0.69	0.76

The Siamese-AlexNet architecture shows the best area under the *ROC* curve ($AUC = 0.844$) with a true and false positive rates of 74,56% and 20,41%, respectively, for a threshold value of 1.56. This yields to an accuracy of 77.07%, a precision equals to 78.50%, and a F_1-score of 0.764. Likewise, the `vgg16` model performance closely follows AlexNet with similar TPRs and slightly lower AUC at 0.827.

AlexNet is commonly embedded into the Siamese Neural Networks architecture for several applications [76]. For this reason, and the performance results in Table 4.1, a few additional experiments are carried out using the AlexNet embedded in the presented framework. First, The Siamese Neural Network architecture embedded with the AlexNet model is fine-tuned and tested using the traditional contrastive loss function and compared to the binary cross entropy loss function, whose results are presented in Table 4.1. Table 4.2 shows comparable results between the tested approaches, the contrastive loss approach shows a similar accuracy score when compared to the BCE but considerably lower precision and F_1-score.

Table 4.2: Performance experiments on SNN-AlexNet using two different loss functions: BCE ($Alexnet$), and contrastive loss ($Alexnet_{CL}$)

Embedded CNN	TPR	FPR	acc	Pr	F1	AUC
$AlexNet$	0.74	0.20	0.77	0.78	0.76	0.84
$AlexNet_{CL}$	0.79	0.24	0.76	0.41	0.11	0.82

Following the proposed experimental approach, the SNN-AlexNet model is used to compare different learning strategies. Figure 4.4 shows *ROC* curves for the SNN-AlexNet architecture when trained from the scratch on the OpenLogo dataset ($AlexNet_s$) and using pre-learned features from ImageNet without further fine-tuning ($AlexNet_{nft}$). The fine-tuned SNN-AlexNet architecture shown in Table 4.1 is added for reference. Table 4.3 shows the scores based on the threshold level with the highest accuracy.

When trained from the scratch the SNN-AleXNet architecture ($AlexNet_s$) achieves an area under the *ROC* curve equals to 0.47. The pre-trained AlexNet embedded in the SNN architecture with no further fine-tuning achieves a considerable higher area under the *ROC* curve (0.74). This experiment shows how

Table 4.3: Performance experiments on SNN-AlexNet with different learning approaches: fine-tuned SNN-AlexNet model ($AlexNet$), SNN-AlexNet model trained from scratch ($AlexNet_s$), pre-trained $SN_{1,2}$ without further fine-tuning ($AlexNet_{nft}$)

Embedded CNN	TPR	FPR	acc	Pr	F1	AUC
$AlexNet$	0.74	0.20	0.77	0.78	0.76	0.84
$AlexNet_s$	0.67	0.44	0.60	0.59	0.63	0.65
$AlexNet_{nft}$	0.60	0.21	0.69	0.73	0.66	0.74

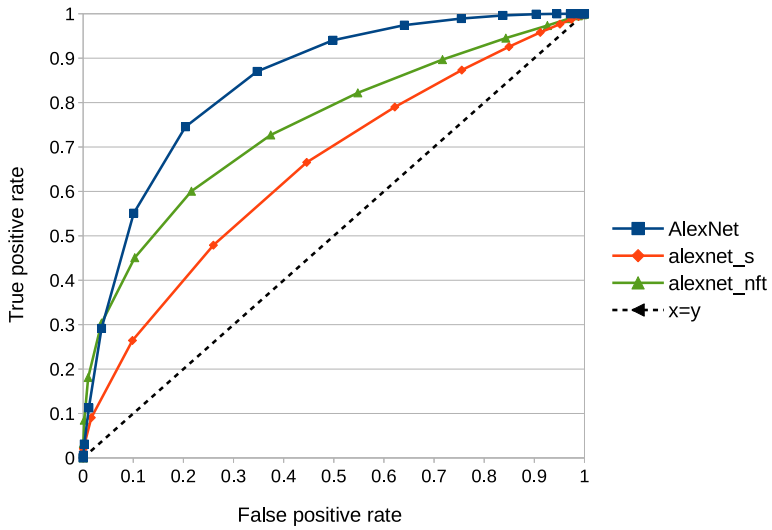


Figure 4.4: *ROC* curve showing the performance of the SNN-AlexNet approach with the introduced training approaches: fine-tuned SNN-AlexNet model ($AlexNet$), SNN-AlexNet model trained from scratch ($AlexNet_s$), pre-trained $AlexNet$ without further fine-tuning ($AlexNet_{nft}$).

the SNN architecture can be highly benefited from prelearned features when tackling the one-shot logo recognition task.

Finally, the SNN- AlexNet architecture is used to test two additional similarity metric layer strategies. First, the SNN-AlexNet architecture is tested by replacing the linear layer of a single output neuron with the Euclidean distance metric D, and in a similar way, replacing the last layer with two output neurons layer defining match and mismatch categories. In particular, when using a linear layer with two neurons as output, it is not possible to draw an ROC curve given that the matching category is assigned directly by the model (see last row of Table 4.4). Again, the fine-tuned SNN-AlexNet architecture shown in Table 4.1

is added for reference. Table 4.4 shows the performance scores achieved by each distance approach.

Table 4.4: Performance experiments on SNN-AlexNet for different output distance approaches: reference fine-tuned SNN-AlexNet model using the linear layer with a single output ($AlexNet$), SNN-AlexNet using the Euclidean distance metric ($AlexNet_D$), SNN-AlexNet using an output layer with two neurons ($AlexNet_{2out}$)

Embedded CNN	TPR	FPR	acc	Pr	F1	AUC
$AlexNet$	0.74	0.20	0.77	0.78	0.76	0.84
$AlexNet_D$	0.57	0.17	0.70	0.76	0.65	0.72
$AlexNet_{2out}$	0.72	0.25	0.74	0.52	0.61	-

The results presented in tables 4.2, 4.3, and 4.4, indicate that the fine-tuned SNN-AlexNet approach using the BCE loss function and the model using the linear layer with a single output offers better results when tackling the one-shot logo recognition problem in the QMUL Open-Logo dataset. A slight decrease in accuracy and area under the ROC curve can be noticed when changing either the loss function, learning approach, or the distance D computation.

Additionally, The pre-trained AlexNet architecture showed some capability to generalise features from the generic object classification task using the ImageNet dataset to the one-shot logo recognition context when embedded into the SNN architecture, achieving a 0.69 accuracy with no training or fine-tuning over the QMUL-OpenLogo dataset. Furthermore, training the proposed approach from scratch produces a considerable impact on the performance with an AUC under 0.5 and low accuracy, precision and $F1$-score, demonstrating the difficulty of training models when an appropriate amount of data is not available.

4.4 Concluding remarks

This chapter presents a study of the Siamese Neural Network approach for logo recognition under one-shot constraints. Although, current state-of-art models for object recognition can achieve accuracy over 90% under very specific conditions, they are rendered obsolete when not enough data is available for training

as in the one-shot recognition task. In this case the use of Siamese Neural Networks for one-shot recognition is presented in relevant literature as a feasible alternative.

Specifically, the chapter evaluates the performance of Siamese Neural Networks for logo recognition with different backbone models. The presented results show that SNNs can achieve accuracy over 75% under adequate experimental conditions. It has to be noticed that the training of such models is not a trivial task, and tools such as transfer learning via fine tuning can be the leading path to appropriate results. In particular, the SNN using an AlexNet backbone showed the best recognition performance closely followed by the vgg Siamese network. This matches the trend found in the literature, where AlexNet is commonly used as the backbone in Siamese networks.

Although the deeper models may perform better in specific domains, when comparing the AlexNet and vgg backbone SNNs the computational cost of training of training the deeper vgg backbone is not reflected in the accuracy performance for one-shot recognition. For these reasons, the proposed recognition approach in the following chapter is based on the AlexNet backbone Siamese Neural Network, which also facilitates a straght forward comparison between the models.

One noticeable characteristic of classic Siamese Neural Networks is that the only comparison operation for similarity matching occurs at the top of the SNN architecture, where distance metric is calculated. This means that the sister networks are trained to generate a small feature vector that is expected to encode the features that discriminate matches and mismatches between the input images. This approach may disregard features encoded within the convolutional layers of the sister network architectures that could contribute to model's performance.

Siamese Neural Networks are specifically designed to address the one-shot object recognition task. Expanding object recognition approaches to engage with object detection is a challenging task, even with enough training data available. The development or adaptation of techniques for multi-box object detection under one-shot detection constraints pose an even greater challenge.

Such approaches would offer desirable advantages for modern computer vision systems, enabling access to a broader spectrum of applications in the future. The following chapter explores how interactions between the branches of a pairwise Siamese Neural Network architecture could potentially improve its performance and enable the development of a one-shot multi-box object detection approach.

Chapter 5

Joint Neural Networks

Relevant literature offers methods that can be adapted to tackle the one-shot object recognition problem and applied to challenging datasets. This chapter focuses on improving one of the perceived weaknesses of the traditional Siamese neural network approach to one-shot recognition and exploring the feasibility of adapting the proposed approach for one-shot object detection.

As stated in the previous chapter, one of the main characteristics in the Siamese architecture is the use of a single layer at the top of the Sister networks to define the similarity between two patterns. For this purpose, each sister network outputs a feature vector that is expected to encode the input patterns. The encoding feature vector size is defined by the number of neurons on the output layers of the sister networks. For instance, the feature vector size of the SNN-AlexNet architecture is equals to 1000; where, the embedded AlexNet architecture is originally designed to classify among that number of categories.

This work proposes the use of a mechanism to share information between the branches of the Siamese architecture in order to allow the model to learn features that define matches and mismatches from lower levels in the architecture. The proposed mechanism consists of adding additional layers to the sister networks architecture that will concatenate outputs from previous layers on both branches in order to further leverage data sharing. For simplicity the sharing data mechanism proposed in this chapter will be called Joint layers and

the corresponding network architecture is referred as Joint Neural Networks (JNNs).

The second part of this chapter addresses the feasibility of adapting the proposed one-shot object recognition approach to the detection case. Modern state-of-art methods for object detection can be classified into one-stage and two-stage approaches.

On one hand, two-stage approaches are the first set of proposed methods for generic object detection using deep learning. Such approaches initially use region proposal techniques like selective search or RPNs, in order to define possible regions with objects and later classify each one of those regions using CNNs. On the other hand, one-stage approaches proposed the use of default bounding boxes to support multibox object detection by passing an input image through the defined models a single time and predicting the confidence of the default bounding boxes to contain objects from specific classes.

The one-shot object detection approach proposed in this chapter is inspired by two-stage multi-box one-shot object detection and the studied Siamese neural networks architecture. Here, the proposed one-shot object recognition approach is adapted to perform multi-box one-shot object detection in a target image for a given query containing a pattern of interest from a unseen category. This method aims to tackle the data limitation that state-of-art approaches require in order to achieve adequate results.

5.1 Joint layers

This section presents the mechanism proposed to share information between the branches of the join neural networks. The so called joint layers are convolutional layers like those found in common convolutional neural networks; where, the feature vector taken as input is the result of a concatenation operation between the previous convolutional layers on each branch of the Joint Neural Network. Figure 5.1 illustrates the structure of a joint layer.

Here $f_i(x^{(1)})$ and $f_i(x^{(2)})$ are the feature vectors outputted by the ith convolutional layer for the first and second branch respectively. $F_j(f_i(x^{(1)}) \oplus f_i(x^{(2)}))$

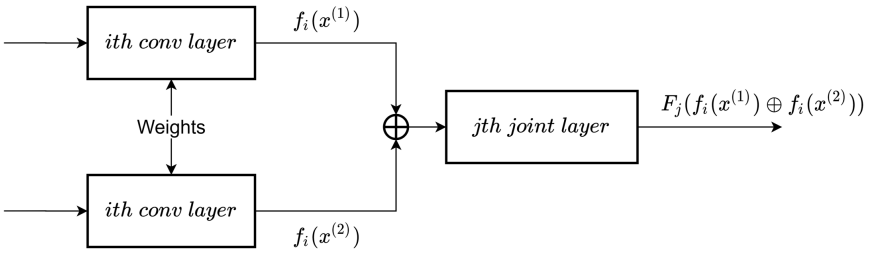


Figure 5.1: Joint layers structure.

is the feature vector outputted by the jth joint layer. And $\oplus$ is the standard concatenation operation merging the output vectors $f_i(x^{(1)})$ and $f_i(x^{(2)})$.

The input size of a joint layer is equals to two times the size of the output feature maps f_i obtained from the previous convolutional layer. The output size of the output feature vector F_j resulting from a joint layer is defined from the architecture design and ideally is fit to match the architecture embedded in the Joint Neural Network.

The joint layers can be embedded as part of any defined architecture such as AlexNet and Darknet as explained in the following sections.

5.2 Joint Neural Networks for one-shot recognition

The proposed Joint Neural Networks are a pairwise model that takes two input images in order to define a similarity metric. The input images are processed in a pair of parallel branches of convolutional neural networks similarly to Siamese neural networks. The JNNs are defined by embedding the previously described joint layers into the a CNN model to support information sharing between the architecture pair of branches.

The initial goal of the proposed JNNs is to perform one-shot pattern recognition specifically for category matching following the problem definition presented in section 1.2; where the defined model takes a pair of images $x^{(1)}$, $x^{(2)}$ as input and outputs a similarity metric S supported by a set of joint layers. A match

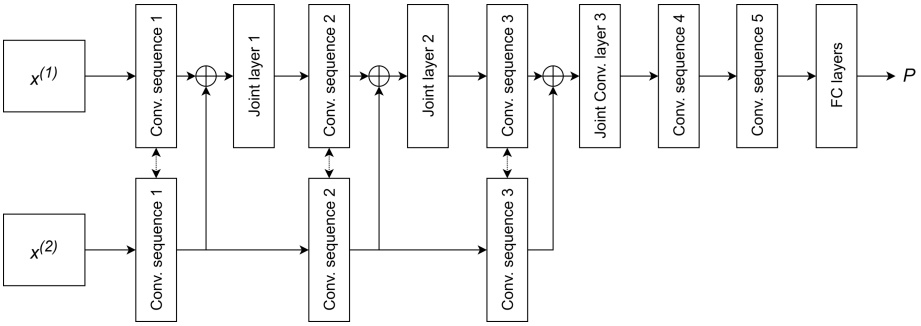


Figure 5.2: Proposed architecture for one-shot object recognition.

or mismatch is defined by applying a threshold t to the similarity metric P.

AlexNet is the commonly preferred architecture to embed in Siamese neural networks for similarity matching [76]. For comparison reasons the proposed Joint Neural Network in this section will adapt the AlexNet architecture as the backbone model to tackle the one-shot recognition problem.

Architecture: Figure 5.2 depicts the proposed architecture showing the two branches of the network and the defined joint layers. The main branch of the architecture at the top of Figure 5.2 is built following the AlexNet design, with the additional joint layers that will integrate outputs from the secondary branch at the bottom. The convolutional layers in the secondary branch of the architecture share weights with the main branch, the same way as sister networks in traditional SNN approaches. The main difference between the Joint Neural Networks and Siamese neural networks is that the secondary branch of the JNNs is not a complete mirror of the main branch. In this way, the JNN model can be trained to define the features that better discriminate similarities and differences between two patterns instead of comparing an squeezed encoding of the input images.

The number of filters outputted by the joint layers is defined so that the next convolutional layer keep the same inputs as defined in the original AlexNet architecture as shown in detail in Table 5.1. Additionally, the output of the similarity metric P is defined by a layer with a single neuron in the same way as the Siamese neural networks initially defined in chapter 4.

Loss function: In contrast to the traditional contrastive loss function this

Table 5.1: Detailed description of the proposed architecture using AlexNet for one-shot recognition

Type	Filters	Size/Stride
Convolutional 1	64	11x11/4
Maxpool		3x3/2
Joint layer 1	64	3x3/1
Convolutional 2	192	5x5/1
Maxpool		3x3/2
Joint layer 2	192	5x5/1
Convolutional 3	384	3x3/1
Joint layer 3	384	3x3/1
Convolutional 4	256	3x3/1
Convolutional 5	256	3x3/1
Maxpool		3x3/2
Linear	4096	
Linear	4096	
Linear	1	

work uses the binary cross entropy (BCE) loss function for training the proposed architecture as defined in equation 4.2. In the same way as Siamese neural networks, the loss function is designed to output a small values for the similarity metric P when the input patterns depicted in the input images $x^{(1)}$ and $x^{(2)}$ are similar; and large values of P when the patterns are dissimilar, i.e.: do not belong to the same category.

Learning: The proposed joint network architecture is trained from scratch and compared to its Siamese network counter part in Table 4.3 ($AlexNet_s$). In order to achieve the one-shot object recognition task as defined in section 1.2, the testing and training categories do not overlap; this is, all the test images remain unseen during the training stage. In the same manner as in chapter 4, the training data loader will select a set of random queries for each epoch with one single query image per batch, matched with random target images that contains the query logo with a 50% probability. Each query and target image is then passed through the model and paired with the ground truth annotations to compute the loss function.

The QMUL Open-Logo [44] dataset is used again for testing the proposed approach. Following the learning protocol proposed in chapter 4 the dataset is

split 60/40 between training and testing categories; this is, 211 categories for training and 125 categories for testing. An additional experiment is conducted by splitting the QMUL-OpenLogo dataset into four different training and testing sets as defined in Appendix A.

As previously mention, the QMUL Open-Logo dataset is a challenging dataset designed for logo detection with little training data. For this reason the dataset must be first cropped using the the ground truth bounding box annotations to extract the regions of interests only, where the patterns used for recognition are located. The training process is iterated for 200 epochs with a learning rate of 0.005 and a batch size of 64. Image pairs are randomly selected from the training set with a 50/50 split between matches and mismatches.

Although the proposed network architecture in this section is adapted from the AlexNet architecture, the joint layers can be embedded into any backbone architecture to define a Joint Neural Network. This makes the Joint Neural Networks an scalable approach that can take advantage of deeper models to perform recognition.

Results of the proposed approach for one-shot recognition are detailed in section 5.4.1. The following section introduces an adaptation of the proposed Joint Neural Networks for one-shot object detection inspired by state-of-art one-stage multibox object detection approaches.

5.3 Joint Neural Networks for one-shot detection

The one-shot object detection problem aims to predict the location of an specific query pattern within a target image. Traditional state-of-art multi-box object detection approaches use a single input image and match all the present patterns to a set of predefined categories, for which the model is trained to recognize and localize.

The main goal of the one-shot object detection approach as proposed in this section is to detect objects in a target image $x^{(1)}$ defined by the category

(a) Query image $x^{(2)}$

(b) Target image $x^{(1)}$

Figure 5.3: One-shot detection task overview. Expected prediction result in red.

depicted by the pattern in the query input image $x^{(2)}$ as shown in Figure 5.3. This is performed complying with the porblem defined in section 1.2, where the category of the query pattern depicted in image the query input image remains unseen during the training process.

Inspired by the one-stage object detection approaches, an anchor window or default box concept is used to define a Joint network architecture for one-shot object detection. Following [8] we split the input target image $x^{(1)}$ in a $S \times S$ grid. Each grid cell predicts B anchor windows with associated coordinates and confidence score $(x, y, w, h, conf)$; where, the output predictions are encoded in a $S \times S \times B \times 5$ tensor. In contrast with [8] and [33] the category to be detected in the target image $x^{(1)}$ is defined by the input query image $x^{(2)}$, so the output tensor does not require an additional positions predict categories as in YOLO.

The same approach and notation as in [33] is used to define the anchor windows and associated predictions within the target image. For each anchor window in the $S \times S$ grid the model outputs five values: t_x, t_y, t_w, t_h, and t_o. The cell offsets from the top left corner of the image are defined as (c_x, c_y) and the bounding box prior has width and height p_w, p_h. The resulting predictions are obtained using equations 5.1 to 5.5 as illustrated in Figure 2.10 from [33].

$$b_x = \sigma(t_x) + c_x \tag{5.1}$$

$$b_y = \sigma(t_y) + c_y \tag{5.2}$$

$$b_w = p_w e^{\sigma(t_w)} \tag{5.3}$$

$$b_h = p_h e^{\sigma(t_h)} \tag{5.4}$$

$$conf = \sigma(t_o) \tag{5.5}$$

Architecture: In order to produce the expected output vector $\hat{P}$ containing the location and confidence predictions for all the predefined anchor boxes, the Joint Neural Network is defined based on the Darknet-19 architecture [33]. Following the same approach to create JNNs for one-shot recognition a few joint layers are added to the backbone architecture.

Figure 5.4 shows the base network structure and Table 5.2 presents the details of each layer. The architecture used for the experimental section will be fyllu defined following the ablation study presented in section 5.4.3.

The architecture defines a Joint Neural Network based on Darknet-19 [33], with 2 input branches as in Siamese Neural Networks. The input image pair is connected to the main and secondary networks sharing weights on the convolutional layers and sharing information via joint layers at three points on the process pipeline. The vector of predictions $\hat{P}$ is given by the output convolutional layer that encodes the locations and confidences in a $S \times S \times B \times 5$ tensor as explained previously; where $S = 14$ for the proposed architecture.

The input sizes for the target and query images are 448x448 and 224x224 respectively. By adapting the architecture to apply the first pooling layer on the target sister network only, the output sizes for the first 2 convolutional layers become even. This small adaptation makes the concatenation operation and further convolutional layers computation straightforward. In addition,the called joint convolutional layers in Figure 5.4 are designed to take the concatenation of the previous convolutional layer outputs for each sister network as input and accommodate its output following the DarkNet19 architecture in the same way as the residual residual layer [33]. Finally, the last convolutional layer outputs a 14×14 tensor composed of $B \times 5$ filters that will encode the model predictions.

Loss function: The loss function used here follows the structure of that presented in [8], where a weighted sum of losses for localization, confidence, and category components is presented. Here two terms are used to build the loss function; the first term describing the location loss, dealing with how close the

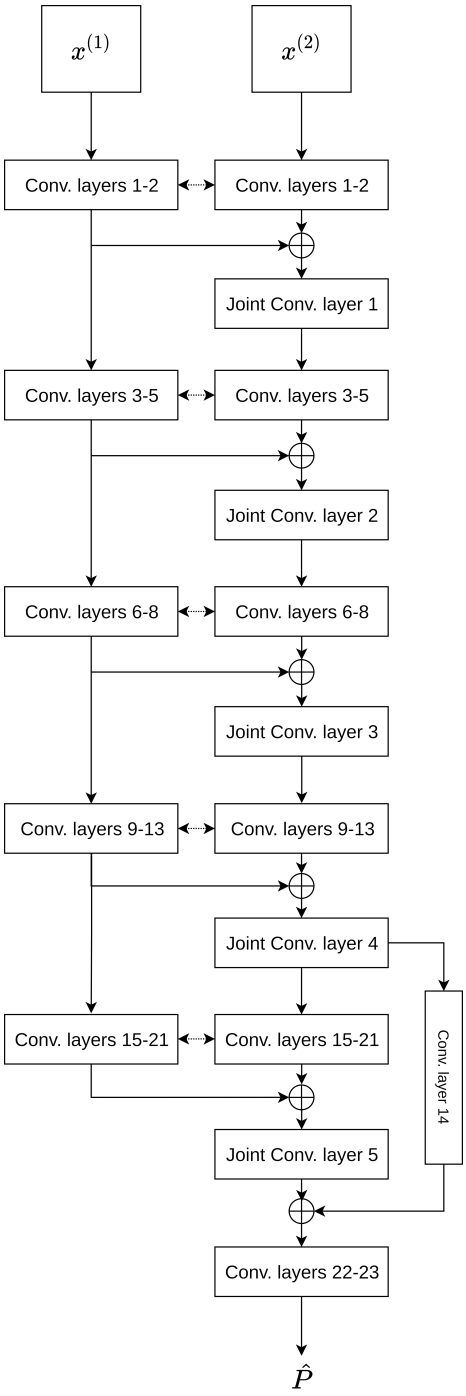


Figure 5.4: Proposed architecture for one-shot detection.

Table 5.2: Detailed description of the proposed architecture.

Type	Filters	Size/Stride	Output
Convolutional 1	32	3x3	224x224
Maxpool		2x2 / 2	112x112
Convolutional 2	64	3x3	112x112
Joint conv. 1	64	3x3	112x112
Convolutional 3	128	3x3	112x112
Convolutional 4	64	1x1	112x112
Convolutional 5	128	3x3	112x112
Maxpool		2x2 / 2	56x56
Joint conv. 2	128	3x3	56x56
Convolutional 6	256	3x3	56x56
Convolutional 7	128	1x1	56x56
Convolutional 8	256	3x3	56x56
Maxpool		2x2 / 2	28x28
Joint conv. 3	256	3x3	28x28
Convolutional 9	512	3x3	28x28
Convolutional 10	256	1x1	28x28
Convolutional 11	512	3x3	28x28
Convolutional 12	256	1x1	28x28
Convolutional 13	512	3x3	28x28
Joint conv. 4	512	1x1	28x28
Convolutional 14	64	1x1	28x28
Maxpool		2x2 / 2	14x14
Convolutional 15	1024	3x3	14x14
Convolutional 16	512	1x1	14x14
Convolutional 17	1024	3x3	14x14
Convolutional 18	512	1x1	14x14
Convolutional 19	1024	3x3	14x14
Convolutional 20	1024	3x3	14x14
Convolutional 21	1024	3x3	14x14
Joint conv. 5	1024	3x3	14x14
Convolutional 22	1024	3x3	14x14
Convolutional 23	$B * 5$	1x1	14x14

predicted anchor box location is to the ground truth annotation; secondly, the confidence loss representing how certain the model is that the pattern of interest is located inside a specific anchor box. The ground truth confidence scores are defined as the intersection over union (IOU) metric between the the predicted

anchor boxes location and the annotated ground truth bounding box. The loss is formally defined as:

$$\ell = \ell_{loc} + \ell_{conf}, \tag{5.6}$$

in which,

$$\ell_{loc} = \lambda_{coord} \sum_{i=1}^{S^2} \sum_{j=1}^{B} \mathbb{1}_{ij}^{obj} (y - y_{pred})^2, \tag{5.7}$$

$$\begin{aligned} \ell_{conf} = \lambda_{obj} \sum_{i=1}^{S^2} \sum_{j=1}^{B} \Big[& \mathbb{1}_{ij}^{obj} c \cdot log(c_{pred}) + \\ & (1 - c) \cdot log(1 - c_{pred}) \Big] + \\ \lambda_{noobj} \sum_{i=1}^{S^2} \sum_{j=1}^{B} \Big[& \mathbb{1}_{ij}^{noobj} c \cdot log(c_{pred}) + \\ & (1 - c) \cdot log(1 - c_{pred}) \Big], \end{aligned} \tag{5.8}$$

where S is the size of the grid; B is the number of anchors defined per grid cell; y and y_{pred} are the ground truth and prediction tensors containing the b_x, b_y, b_w, b_h values; c and c_{pred} are the ground truth and predicted confidences ($conf$); $\mathbb{1}_{ij}^{obj}$ is equals to 1 where the anchor window j at cell i has $IOU > 0.5$ with the target pattern and 0 otherwise; $\mathbb{1}_{ij}^{noobj}$ is the complement of $\mathbb{1}_{ij}^{obj}$; and λ_{coord}, λ_{obj}, and λ_{noobj} are weights to deal with the relevance of each term in the loss function. In contrast to [8] we use the Binary Cross Entropy function for the confidence terms along with the λ_{obj}, and λ_{noobj} weights to deal with the learning priority when keeping the true positives or discarding false positives. The data loading for learning follows the same protocol as in recognition; where the target has a 50% chance of containing the query pattern.

Learning: The proposed joint network architecture is trained from scratch and tested over the VOC, COCO, and QMUL Open-Logo dataset, ensuring non-overlapping training/testing categories to comply with the one-shot detection

constraints for unseen categories. The data loader is coded to create image pairs with a matching chance of 50%.

Specifically for the one-shot logo detection use case, the QMUL Open-Logo dataset does not require any cropping or pre-processing. The training process is iterated for 300 epochs, stopping after 20 sucesful iteration without improvement. A learning rate of 0.005 and a batch size of 64 randomly selected image pairs are defined. The proposed Joint Neural Network implementation uses data augmentation to improve detection results as commonly implemented in traditional one-stage object detection approaches. In this way, image transformations are applied to each loaded sample; including color adjustment in the HSV color space, scaling, rotation and translation transforms. Implementation details are specified in the experimental section and the github repository.

The following section presents the one-shot object recognition and detection experiments for the proposed Joint Neural Networks architecture.

5.4 Experiments

The experimental section is split into two subsections: First, evaluating the performance of the one-shot object recognition results proposed in section 5.2. Then, the adaptation of Joint Neural Networks for one-shot object detection from section 5.3.

5.4.1 Recognition

Initially, the architecture proposed in section 5.2 (Figure 5.2) is assessed for the one-shot recognition task as defined in section 1.2. The presented experimental approach consists on training the proposed model with a set of classes and afterwards testing the matching/recognition performance of the model over unseen classes. For this purpose the QMUL-OpenLogo and MiniImageNet dataset described in sections 2.3.2 and 2.3.3 are used. The datasets are divided into training and testing subsets with no overlapping classes following the one-shot problem definition in section 1.2.

First, the experimental settings presented in section 4.3 are reproduced over the QMUL-OpenLogo to compare the proposed approach to Siamese Neural Networks when training from the scratch. In addition the same experiment is conducted over the QMUL-OpenLogo dataset for four different training and testing data splits. Then, the proposed approach is compared to state-of-the art one-shot learning algorithms for object recognition over the MiniImageNet dataset following the experimental settings defined in [67]

In order to evaluate the performance of the proposed approach for one-shot recognition, results are presented using true positive rates, false positive rates, accuracy, precision, and the ROC curve. The proposed approach will be compared with the traditional SNN approach using the AlexNet architecture backbone. Given that different thresholds over the learned similarity metric P lead to differences in the performance of the binary classifier, a set of 20 thresholds is used to the similarity metric within the range of responses returned by the Siamese neural network (0, 1). The source code used to perform the recognition experiments can be found at `https://github.com/cjvargasc/JNN_recog/`.

QMUL-OpenLogo dataset experiments

The proposed Joint Neural Network using AlexNet as the backbone of the architecture is trained with $epochs = 200$, $lr = 0.005$ and $batch\_size = 64$. The training process was around 9 hours to complete on a GeForce GTX 1080. The proposed approach as well as the Siamese AlexNet architecture used for experiments are implemented using the pythorch library.

Figure 5.5 shows the performance of the proposed approach under the same experimental settings as in section 4.3, for logo recognition in terms of the *ROC* curve obtained by thresholding the similarity metric P. Both models are trained from the scratch for simplicity and fair comparison. The accuracy, precision and area under the ROC curve (AUC), reported for each method are measured for the threshold with the highest accuracy.

Table 5.3 shows the detailed result scores for the approaches in the *ROC* curve (Figure 5.5). The proposed approach shows a significant improvement over the AUC, FPR and accuracy scores when compared with the standard

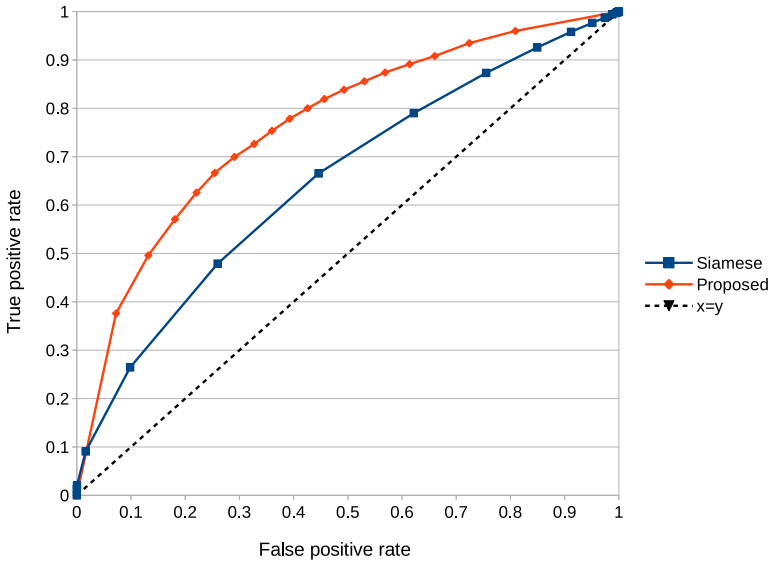


Figure 5.5: *ROC* curve showing the performance of the proposed approach and the standard SNN approach on the QMUL-OpenLogo dataset.

SNN AlexNet approach ($AUC = 0.76$, $FPR = 0.25$, $acc = 0.70$).

Additionally, Table 5.4 presents the assessment of the proposed approach for 4 different training and testing data splits as defined in Appendix A. Results show that the proposed approach outperforms the traditional SNN approach in all the proposed data splits in terms of accuracy and precision.

Figure 5.6 depicts the predicted value P for a few arbitrarily selected logos unseen during training. Although, the QMUL-OpenLogo renders a challenging scenario for both object recognition and detection, to the best of our knowledge it has not been used in the context of one-shot object recognition. For this reason, within this set of experiments the proposed approach is compared against the baseline presented in chapter 4.

Table 5.3: Performance metrics for the proposed JNN approach and SNN AlexNet trained from scratch using the QMUL-OpenLogo dataset.

Method	TPR	FPR	acc	Pr	AUC
JNN	0.66	0.25	**0.70**	**0.72**	**0.76**
SNN	0.67	0.44	0.60	0.59	0.65

Table 5.4: Performance metrics for the proposed approach and SNN AlexNet in four different training and testing data splits.

Metric	split 1		split 2		split 3		split 4	
	JNN	SNN	**JNN**	SNN	**JNN**	SNN	**JNN**	SNN
TPR	0.66	0.75	0.61	0.59	0.64	0.59	0.68	0.57
FPR	0.43	0.64	0.30	0.41	0.13	0.33	0.25	0.35
accurracy	**0.61**	0.55	**0.65**	0.59	**0.76**	0.63	**0.72**	0.61
Precision	**0.60**	0.54	**0.67**	0.59	**0.83**	0.64	**0.73**	0.62
F1score	0.63	0.63	**0.64**	0.58	**0.72**	0.61	**0.70**	0.60

(a) $P = 0.63$ (b) $P = 0.53$

(c) $P = 0.13$ (d) $P = 0.15$

Figure 5.6: One-shot recognition results for arbitrarily selected pairs of images from the QMUL-OpenLogo and MiniImageNet datasets.

MiniImageNet dataset experiments

The same implementation and hardware from the previous experiments was used to trained the model in the more generic object recognition dataset miniImageNet described in section 2.13. The miniImageNet proposes a data split complying with the one-shot object recognition constraints, where training and testing classes do not overlap. Initially, the miniImageNet results are presented for the SNN AlexNet introduced in chapter 4 and the proposed approach, given that the same ROC analysis can be performed at several threshold levels. Then, the proposed approach is compared to the state-of-art methods for one-shot learning.

Figure 5.7 shows the performance of the proposed approach for object recognition using the *ROC* curve, obtained by thresholding the output similarity metric S. For both models, the similarity metric is thresholded in the range of responses of the model to compute and plot the true positive (TPR) and false

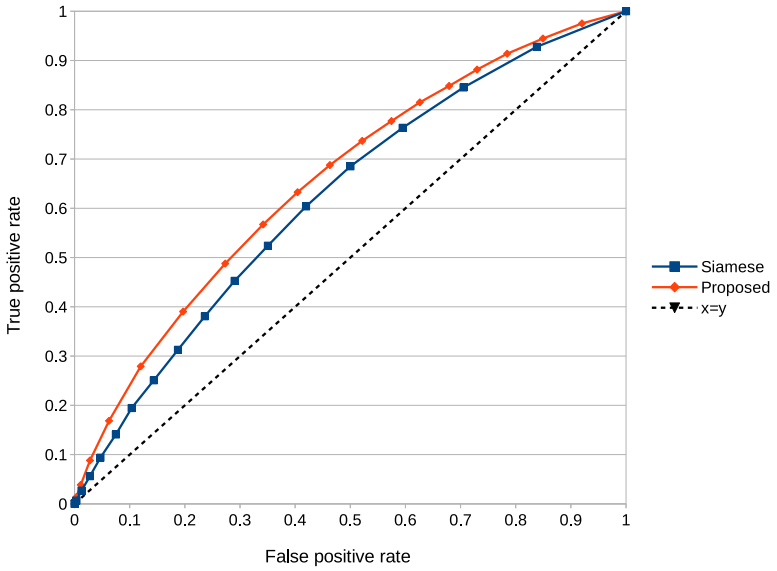


Figure 5.7: *ROC* curve showing the performance of the proposed approach and the standard SNN approach on the miniImageNet dataset.

positive (FPR) rates. The accuracy, precision, and area under the ROC curve (AUC), reported for each method are measured for the threshold resulting in the highest accuracy.

Table 5.5 shows the detailed result scores for the approaches in Figure 5.7. The proposed approach shows a small performance improvement over the AUC, FPR and accuracy scores when compared with the standard SNN AlexNet approach.

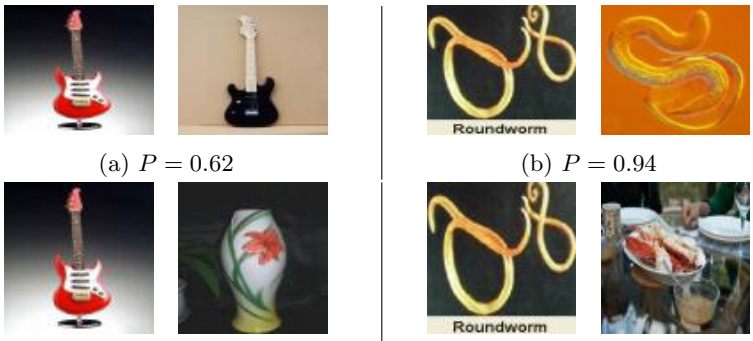
Table 5.5: Performance metrics for the proposed JNN approach and SNN AlexNet trained from scratch using the MiniImageNet dataset.

Method	TPR	FPR	acc	Pr	AUC
Proposed	0.63	0.40	0.614	0.61	0.65
Siamese	0.60	0.41	0.592	0.58	0.62

Table 5.6 shows the accuracy scores for state-of-art one-shot object recognition approaches assessed in miniImageNet. The proposed approach shows a small increase in accuracy from the Meta-Transfer learning for few-Shot learning (MTL) presented in [71]. This increase is achieved without the pre-learned knowledge used in the MTL approach leaving room for possible improvement.

Table 5.6: Accuracy measured for the proposed approach compared to state-of-art approaches for one-shot recognition on the miniImageNet dataset.

Method	accuracy (%)
MALM [67]	48.70
ProtoNets [68]	49.42
SNAIL [69]	55.71
TADAM [70]	58.50
MTL [71]	61.20
Proposed	**61.41**



(a) $P = 0.62$ (b) $P = 0.94$

(c) $P = 0.23$ (d) $P = 0.18$

Figure 5.8: One-shot recognition results for arbitrarily selected pairs of images from the QMUL-OpenLogo and MiniImageNet datasets.

Figure 5.8 depicts the predicted value P for a few arbitrarily MiniImageNet categories unseen during training. The presented results on both QMUL-OpenLogo and miniImageNet datasets demonstrate that the proposed Joint Neural Networks are a competitive approach for one-shot object recogntion.

5.4.2 Detection

The proposed JNN based detection approach is tested under four experimental settings using the QMUL-OpenLogo, VOC, and COCO datasets. Firstly, the experimental settings proposed in [95] are used to measure the introduced JNN performance and compare it to state-of-art approaches. Each model is trained using 16 of the 20 classes of the VOC dataset and tested on the other 4 classes. The model is trained using the VOC 2007 and 2012 trainval set and evaluated us-

ing the VOC 2007 test set. Secondly, the proposed approach is tested by training the model using the COCO dataset and testing with the VOC dataset, as introduced in [96]. During training, the COCO dataset classes that overlap the VOC dataset are excluded, in order to comply with the one-shot constraints. Thirdly, the proposed approach is trained and tested using the COCO dataset only. The dataset is divided in four different training and testing sets as introduced in [97] (see Appendix B). Finally, the model is tested on the QMUL-OpenLogo dataset using a 60/40 split for training and testing classes as presented in the recognition experiments and complying with the one-shot constraints. The presented results for the state-of-art approaches used for comparison are taken directly from the cited authors work following their availability.

The proposed JNN is trained using the SGD optimizer with a learning rate equals to 0.0001, a 0.9 momentum for 160 epochs, with a 16 batch size, over a single GeForce GTX 1080 GPU. All the experiments are performed using an IoU threshold of 0.5. The source code used to perform the recognition experiments can be found at `https://github.com/cjvargasc/JNN_detection/`.

Table 5.7 shows the results for the first experimental setting, along with the train and test class splits for the VOC dataset as in [95]. Under this experimental settings, results show a 13% mAP difference to the top performing approach [96]. However, it should be noted that in contrast to the results presented from the previous approaches we do not pre-train our model. The results table presented in [96] use the pretrained models on the full ImageNet dataset where the test classes may be present, meaning that the one-shot constraint is not in strict use.

The authors in [96] go further experimenting with one-shot object detection, using the COCO and VOC datasets. Following their approach, the second experimental setting presented for one-shot object detection trains the proposed model on the COCO dataset excluding the VOC classes used for testing. Table 5.8 shows the results compared to the M-FCOS approach [96]. Results show a mAP improvement using the JNN based detection JNN based detection approach presented in this paper when compared to the top performing approach [96] in Table 5.7.

In addition, the performance of the JNN is also compared to the state-of-art

Table 5.7: Comparison of different one-shot detection methods on VOC in AP (%). Note that SiamFC, SiamRPN, and CompNet use all classes in their ImageNet pre-trained backbones.

Method	Seen classes																	Unseen classes				
	plant	sofa	tv	car	bottle	boat	chair	person	bus	train	horse	bike	dog	bird	mbike	table	mAP	cow	sheep	cat	aero	mAP
SiamFC [98]	3.2	22.8	5.0	16.7	0.5	8.1	1.2	4.2	22.2	22.6	27.8	35.4	14.2	25.8	11.7	19.7	15.1	6.8	2.28	31.6	12.4	13.3
SiamRPN [97]	1.9	15.7	4.5	12.8	1.0	1.1	6.1	8.7	7.9	6.9	5.1	17.4	17.8	20.5	7.2	18.5	9.6	15.9	15.7	21.7	3.5	14.2
CompNet [99]	28.4	41.5	65.0	66.4	37.1	49.8	16.2	31.7	69.7	73.1	39.8	75.6	71.6	61.4	52.3	63.4	52.7	75.3	60.0	47.9	25.3	52.1
CoAE [95]	30.0	54.9	64.1	66.7	40.1	54.1	14.7	60.9	77.5	78.3	46.2	77.9	73.2	80.5	70.8	72.4	60.1	83.9	67.1	75.6	46.2	68.2
M-FCOS [96]	33.7	58.2	67.5	72.7	40.8	48.2	20.1	55.4	78.2	79.0	48.8	76.2	74.6	81.3	71.6	72.0	**61.1**	74.3	68.5	81.0	52.4	**69.1**
JNN	4.5	44.0	38.3	51.5	2.9	43.76	13.1	39.3	45.2	57.8	72.6	58.0	75.5	64.2	71.14	12.7	43.4	64.7	51.0	65.2	43.5	56.1

Table 5.8: Comparison of different one-shot detection methods on VOC in AP (%). Note that SiamFC, SiamRPN, and CompNet use all classes in their ImageNet pre-trained backbones..

Method	plant	sofa	tv	car	bottle	boat	chair	person	bus	train	horse	bike	dog	bird	mbike	table	cow	sheep	cat	aero	mAP
M-FCOS [96]	10.3	79.1	54.2	64.0	18.4	19.6	26.3	9.6	67.5	55.4	67.7	26.8	73.7	50.8	26.4	11.9	84.8	75.1	69.5	28.4	45.9
JNN	9.5	69.3	49.8	60.3	7.2	29.1	10.1	6.7	60.3	57.2	58.5	45.3	62.6	45.6	74.8	29.0	70.4	55.4	54.4	88.1	**47.1**

Table 5.9: Evaluation on COCO with respect to mAP score (%) for unseen classes.

split	1	2	3	4	Average
SiamRPN [97]	15.3	17.6	17.4	17.0	16.8
coAE [95]	23.4	23.6	20.5	20.4	**22.0**
JNN	12.9	16.5	10.0	14.4	13.5

approaches using the experimental approach presented in [97] and [95]. In this case, four different splits of training and testing categories are defined over the COCO dataset. The fully detailed category splits can be found in Appendix B. Each training split will contain 60 categories leaving 20 for testing. Table 5.9 shows the results of the proposed approach compared to the SiamRPN [97] and coAcoE [95] approaches. Under this experimental setting our approach is down 8.5% mAP when compared with the CoAE approach [95]. The coAE approach [95] shows an average mAP of 22.0%, rendering this experimental setting as a challenging task for one-shot object detection approaches.

Finally, the proposed approach is assessed in the context of one-shot logo detection using the QMUL-OpenLogo dataset. Here, 211 categories are arbitrarily selected for training and 125 for testing in the same way as the recognition experiments in section 5.4.1. Table 5.10 shows the average precision (AP50) results for the top performing 15 unseen classes and the mean average precision (mAP) for the all 125 unseen classes used for testing. Figure 5.9 show some arbitrarily selected results of the proposed approach in action.

5.4.3 Ablation study

This thesis introduces Joint Neural Networks as an adaptable framework that can be used with any defined backbone architecture to perform the one-shot object recognition or detection tasks, where the joint layers are the key concept used build the Joint Neural Networks. This section explores how the use of joint layers on specific locations on the architecture may affect the performance of the proposed approach.

Figure 5.4 presented in section 5.3 shows the proposed convolutional and joint layers used to tackle the one-shot object detection task. In order to test the

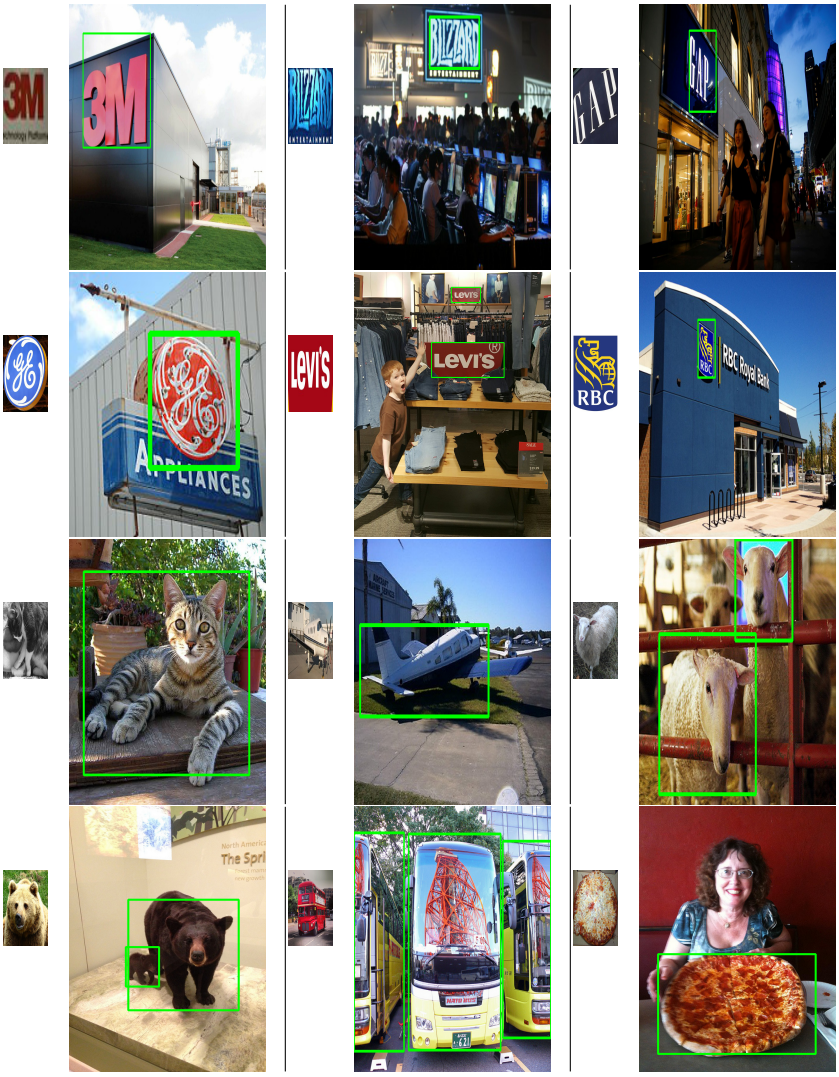


Figure 5.9: One-shot detection results results for arbitrarily selected query and target images from the QMUL-OpenLogo dataset.

Table 5.10: AP detection results for the top 15 Open-Logo dataset classes and the mAP results for the whole dataset.

Class	mAP/AP
anz_text	100.00%
rbc	98.86%
blizzardentertainment	98.34%
costco	93.26%
3m	90.61%
bosch_text	90.00%
gap	89.47%
lexus	88.92%
generalelectric	83.32%
hp	82.81%
levis	79.36%
airhawk	79.17%
danone	79.02%
armitron	77.73%
google	77.66%
all	**52.84**%

firstly proposed architecture, Table 5.11 shows the performance of the proposed framework under different joint layer configurations. The test are carried out using the first experimental settings described in section 5.4.2 (Table 5.7).

Each Joint Layer (JL) in Figure 5.4 is depicted as a column in Table 5.11. The experiments explore the performance of the architecture when adding joint layers sequentialy as well as for groups of two and three layers. Results show that for the one-shot object detection task using the VOC dataset the combination of the joint layers 1, 2, and 4 in DarkNet19 architecture yields the best overall mAP performance (56.1%).

5.5 Concluding remarks

This chapter presents a novel convolutional neural network based architecture named Joint Neural Networks. Joint Neural Networks are inspired by Siamese Neural Networks for one-shot object recognition, aiming to improve the way information is shared between branches when comparing a pair of input im-

Table 5.11: Ablation study for used joint layers in DarkNet19 architecture for detection

JL1	JL2	JL3	JL4	JL5	mAP
×					53.3
×	×				53.6
×	×	×			54.9
×	×	×	×		55.2
×	×	×	×	×	36.8
×			×		52.9
	×		×		53.9
		×	×		47.9
×	×		×		**56.1**
×		×	×		53.7

ages, leveraged by the Joint layer concept. This approach is proposed for the one-shot object recognition and detection tasks where traditional deep learning approaches fall short due the lack of training data available.

The Joint Neural Network approach for one-shot recognition shows better accuracy when compared to the classical Siamese Neural Network approach. Although the amount layers of differ between the approaches, results in chapter 4 indicate that the deepness of the embedded architecture is not necessarily a deciding performance factor when evaluating pairwise architectures. This fact highlights the impact of the Joint layers in the performance of pairwise architectures for one-shot object recognition.

Having defined Joint Neural Networks and inspired by one-stage object detection approaches, this chapter extends the presented JNN architecture to also address the one-shot object detection task. The adaptation of Joint Neural Networks for detection is proposed following the anchor window and confidence prediction paradigm introduced by one-stage object detection approaches. Detection results present the proposed JNN architecture as a competitive approach for one-shot object detection, outperforming the state-of-art results under some of the evaluated experimental settings.

The presented Joint Neural Networks can be seen as a framework designed with the most basic building blocks of Convolutional Neural Networks and inspired by successful object recognition and detection approaches. In that sense, the same approach can be used to define new architectures and apply well known

machine learning tools such as transfer learning to push the state-of-art performance for one-shot object recognition and detection.

Despite recent progress achieved to address the one-shot object recognition and detection tasks, results indicate that there is a long way before pairwise models can achieve one-shot object recognition or detection results comparable to those of state-of-art approaches where training data is available. The last chapter of this thesis presents the conclusions of this research along with a future work discussion.

Chapter 6

Conclusions and Future Work

Data plays a very important role in the development models that can address computer vision's challenging tasks such as object recognition and detection. Consequently, most of modern approaches that have gain attention in industrial and academic fields require substantial amount of data in order to achieve results comparable to those in the state-of-art literature. This fact stands as a limiting factor when building computer vision systems for real world applications; where, the amount of data required to train state-of-art models is not available or the data compilation costs defeat the purpose of the system.

The one-shot recognition and detection problems define the framework in which methods proposed in the literature may address the previously stated data issues. Early approaches used image processing techniques to extract features from the patterns of interest, later used to define matches between patterns. Such approaches can be used successfully in very constrained test sets, but when applied in challenging datasets with high variability between the test classes and surrounded by complex scenes the performance drops. That is the case of the SURF approach, showing a 9.8% mAP when tested on the QMUL-OpenLogo dataset. For this reason, the academic community has turned into more sophisticated approaches to encode the variability of patterns of interest

to perform object recognition.

With the introduction of convolutional neural networks the requirement for bigger datasets became obvious, and approaches for data syntheses and transfer learning gained considerable attention. These methods have been successfully applied in several use cases, but the one-shot constraints still present a challenge when such approaches are applied for complex tasks. In other words, the one-shot problem has been successfully addressed in the literature using plain data syntheses or transfer learning for tasks like character recognition, in some cases utilizing additional information from context is employed e.g.: stroke direction when drawing a character. But when a complex pattern such as a logo has to be generated from a single sample, the problems related to the one-shot constraint become evident.

Siamese neural networks appear in the literature as a feasible alternative to face the one-shot recognition problem and remain as an active topic in several application fields such as object tracking, person re-identification, and face recognition. The main characteristic that makes Siamese neural network an appealing research approach is the fact that is designed to compute a similarity metric between two inputs instead of the traditional category definition for a fixed set of classes. This research shows that Siamese Neural Networks can achieve up to 77% accuracy in the case of one-shot logo recognition when trained properly. But also that training such approach is not a trivial task, and the accuracy can drop to 60% when for instance no pre-training is used. This approach highly depends on the capability of the sister networks to encode the information of the input patterns into feature vectors, limited by the size of the encoding.

In this way, the proposed Joint Neural Networks offer an alternative and feasible solution for one-shot object recognition and detection by leveraging the interaction between the branches of the pairwise model via joint layers. When applied for one-shot recognition, Joint Neural Networks showed an accuracy improvement up to 13% compared with Siamese Neural Networks. In addition, the performance of Joint Neural Networks is comparable to state-of-art one-shot learning approaches, without performing any training or fine-tuning over

the unseen tested classes.

In addition, the proposed Joint neural network adaptation for one-shot object detection is the first one-stage detection approach developed to comply with one-shot constraints. The approach shows performance comparable to recent state-of-art one-shot object detection methods, even outperforming the top approach by 1% mAP when training with the COCO dataset and testing on the Pascal VOC dataset. Given that the proposed approach for one-shot object detection is based on early principles of mainstream Convolutional Neural Networks for object detection there is still plenty room for improvement.

This thesis presents in chapter 2 a literature review on one-shot object recognition and detection. Then, chapter 3 proposes a modification of the Viola-Jones algorithm to ease its application for one-shot object detection, along with a performance assessment of image processing approaches for the same task. Chapter 4 studies the capability of Siamese Neural Network models to engage with the one-shot object recognition task in a complex dataset using different backbone models and training configurations. Finally, chapter 5 introduces the Joint Neural Network framework, along with two specific architectures to address the one-shot object recognition and detection tasks.

6.1 Future work

The Joint Neural Networks proposed in this thesis is presented as a competitive approach for one-shot object recognition and detection. Is only until recently that the one-shot object detection problem has been addressed in the relevant literature and there is still a long way to achieve a performance comparable to mainstream object detection approaches, where data is available. In consequence, tasks such as developing a testing framework adopted through the research community to measure the performance of proposed approaches is still a pending task.

In particular for the proposed approach, the adaptation of a feature visualizing mechanism for Joint Neural Networks to better understand the inner characteristics that support pattern matching, may allow a deeper understand-

ing of the training process and method limitations when dealing with one-shot object recognition and detection. In this way, leading to new architectures and effective application of tools to impact positively the framework performance.

Furthermore, the proposed approach is designed using basic building blocks of Convolutional Neural Networks and inspired by early one-stage object detection approaches, as it is the case of YOLO in its second version. This implies that in theory the proposed Joint Neural Networks framework for one-shot object detection can still be expanded by following improvements already proposed for object detection approaches in the YOLO family, such as using different backbone architectures, larger batch sizes, Dropblocks, IOU Loss, and pretrained models.

Finally, a few difficulties can be addressed as future work related to model parameter definition were faced when applying the proposed approach for one-shot object detection. Firstly, the definition of fixed anchor windows impacts the performance of the proposed approach for datasets where the target patters show high size and shape variance, as it is the case of the QMUL-OpenLogo dataset. An study of the trade-off between the execution time related to the definition of several anchor windows versus the capability of those anchor windows to bound the target patterns may lead to improved results in such cases. Secondly, the proposed approach defines a single global confidence threshold to define pattern matches along the entire test dataset. Subjective observed evidence seems to indicate that the optimal confidence threshold may vary between query patterns. Properly defining this fact and the further adaptive definition of the confidence threshold to determines matches can also lead to improve model performance of future JNN versions. Lastly, the model hyper-parameters selected for the proposed approach are defined following the model implementations that inspired them; those being, the SNN-AlexNet and the YOLO darknet. Although, this makes the models comparable for instance in the one-shot object recognition experiments, a deep study into tuning these parameters can improve the performance of the proposed approach.

Appendices

Appendix A

Data splits for one-shot object recognition experiments

This appendix presents the testing data splits used for the one-shot object recognition experiments in section 5.4.1. For that purpose the QMUL-OpenLogo dataset divided into four different test sets, defined over the available classes in the dataset. Each test set has 84 classes for testing, leaving 252 classes available for training the models. Table A.1 shows the testing classes defined for each split.

The testing splits are selected using a randomized approach over a stack built from the dataset class labels. The code to generate the test splits definitions can be found at `https://github.com/cjvargasc/JNN_recog/`.

Table A.1: Category splits of QMUL-OpenLogo.

split 1	split 2	split 3	split 4
pampers	erdinger	armitron	volkswagen
chevron	bottegaveneta	kfc	bridgestone_text
kitkat	aluratek	uniqlo	millerhighlife
bik	standard_liege	pizzahut	target
cvs	aspirin	budweiser	heineken
kelloggs	schwinn	hp	at_and_t
superman	nivea	toyota_text	unitednations
americanexpress	audi	esso_text	boeing
nike	ebay	heraldsun	costco
hyundai	motorola	tissot	jacobscreek
coke	wordpress	homedepot	cvspharmacy
ferrari	goodyear	tsingtao	airhawk
tommyhilfiger	soundrop	mercedesbenz	ups
warnerbros	unicef	nb	allianz
jagermeister	fosters	3m	chiquita
playstation	starbucks	nvidia	select
jackinthebox	toyota	target_text	allianz_text
homedepot_text	bmw	ibm	aral
bankofamerica_text	cpa_australia	chevrolet_text	allett
head_text	texaco	wii	santander
dhl	stellaartois	ec	base
neg_ss	visa	microsoft	kodak
apple	tnt	guinness	lego
michelin	bulgari	underarmour	basf
huawei_text	walmart	aldi_text	spar
bbc	jurlique	adidas_text	skechers
mk	cocacola	mini	reebok1
bellodigital_text	coach	northface	batman
barclays	subaru	bankofamerica	nissan
fly_emirates	burgerking	walmart_text	maxxis
tacobell	airness	londonunderground	shell_text
gillette	republican	google	nissan_text
mobil	rbc	lacoste_text	supreme
comedycentral	hyundai_text	anz_text	ikea
spiderman	citroen	ruffles	bfgoodrich
nbc	bacardi	redbull_text	citroen_text
poloralphlauren	porsche_text	gucci	becks
apc	mcdonalds	vodafone	hermes
benrus	drpepper	hh	rolex_text
total	bellataylor	subway	gildan
teslamotors	abus	bionade	bershka
jcrew	asus	hm	bellodigital
bem	siemens	luxottica	disney
amcrest	honda_text	suzuki	lv
carlsberg	honda	medibank	boeing_text
intel	reeses	obey	netflix
hisense	firefox	mastercard	chickfila
reebok	bbva	colgate	barbie
windows	chanel_text	fedex	fritolay
sprite	hsbc_text	burgerking_text	johnnywalker
carters	loreal	singha	velveeta
facebook	mercedesbenz_text	mccafe	gap
marlboro_fig	aldi	firelli	uniqlo1
nasa	renault	sega	lamborghini
danone	accenture	recycling	tostitos
redbull	opel	santander_text	audi_text
oracle	fritos	marlboro	timberland
yonex	armani	corona_text	sony
espn	rolex	anz	android
t-mobile	generalelectric	converse	hsbc
reebok_text	milka	wellsfargo	chevrolet
prada	citi	blackmores	paulaner
axa	jello	umbro	optus_yes
cartier	caterpillar	canon	adidas1
bayer	youtube	sap	olympics
samsung	aquapac_text	esso	vaio
blizzardentertainment	pizzahut_hut	kia	heineken_text
thomsonreuters	williamhill	shell	puma
volkswagen_text	ford	lexus	bosch_text
costa	volvo	mcdonalds_text	mtv
pepsi	mitsubishi	soundcloud	hanes
yonex_text	panasonic	philips	maserati
zara	cheetos	huawei	levis
us_president	scion_text	verizon	doritos
dunkindonuts	wellsfargo_text	calvinklein	yahoo
carglass	lg	marlboro_text	lacoste
alfaromeo	cisco	adidas	shell_text1
chanel	athalon	hersheys	quick
head	americanexpress_text	infiniti	yamaha
amazon	corona	optus	internetexplorer
apecase	lays	xbox	puma_text
kraft	chimay	nintendo	bosch
rittersport	porsche	budweiser_text	lotto
evernote	nescafe	dexia	bridgestone

Appendix B

COCO data splits for one-shot object detection experiments

This appendix presents the data splits using for training and testing the one-shot object detection approach presented in section 5.4.2. The data splits follow the experimental settings proposed by [97].

This approach splits the 80 object categories in MS-COCO into 60 training and 20 test categories, including every fourth category into the test split starting with the first, second, third or fourth category, respectively. These splits are shown in Table B.1 [97].

Providing four splits with equally distributed held-out categories has two main advantages: It allows to test on all categories in MS-COCO (albeit with different models) while sub sampling the super categories as evenly as possible. This approach assumes that there is some knowledge available from all broad object categories, allowing the model to perform inferences over the unseen categories [97].

Table B.1: Category splits of MS-COCO.

split 1	split 2	split 3	split 4
1 Person	2 Bicycle	3 Car	4 Motorcycle
5 Airplane	6 Bus	7 Train	8 Truck
9 Boat	10 Traffic light	11 Fire Hydrant	12 Stop sign
13 Parking meter	14 Bench	15 Bird	16 Cat
17 Dog	18 Horse	19 Sheep	20 Cow
21 Elephant	22 Bear	23 Zebra	24 Giraffe
25 Backpack	26 Umbrella	27 Handbag	28 Tie
29 Suitcase	30 Frisbee	31 Skis	32 Snowboard
33 Sports ball	34 Kite	35 Baseball bat	36 Baseball glove
37 Skateboard	38 Surfboard	39 Tennis racket	40 Bottle
41 Wine glass	42 Cup	43 Fork	44 Knife
45 Spoon	46 Bowl	47 Banana	48 Apple
49 Sandwich	50 Orange	51 Broccoli	52 Carrot
53 Hot dog	54 Pizza	55 Donut	56 Cake
57 Chair	58 Couch	59 Potted plant	60 Bed
61 Dining table	62 Toilet	63 TV	64 Laptop
65 Mouse	66 Remote	67 Keyboard	68 Cell phone
69 Microwave	70 Oven	71 Toaster	72 Sink
73 Refrigerator	74 Book	75 Clock	76 Vase
77 Scissors	78 Teddy bear	79 Hair drier	80 Toothbrush

References

[1] A. Jain, R. P. W. Duin, Jianchang Mao: Statistical pattern recognition: a review. in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, pp. 4–37, Jan 2000.

[2] Lasie project website, http://www.lasie-project.eu/

[3] Google lens website, https://lens.google.com/

[4] K. Chandramouli, P. Kudumakis, and E. Izquierdo, International Organisation for Standardisation Organisation Internationale De Normalisation ISO/IEC JTC1/SC29/WG11 Coding of Moving Pictures and Audio ISO/IEC JTC1/SC29/WG11 MPEG2016/ m38557. 30-May-2016.

[5] Choe, J., Park, S., Kim, K., Hyun Park, J., Kim, D., Shim, H. Face generation for low-shot learning using generative adversarial networks. In: ICCV workshop. 2017

[6] A. Antoniou, A. Storkey, H. Edwards: Data augmentation generative adversarial networks. ArXiv preprint arXiv:1701.04340, 2017.

[7] A. Nguyen, J. Yosinski, J, Clune.: Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images. ArXiv e-prints, December 2014.

[8] Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR. 2016.

[9] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed, SSD: Single shot multibox detector, arXiv:1512.02325, 2015.

[10] Lowe, David G. SIFT Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision, Volume 60 Issue 2, November.

[11] Lowe, David G. Object Recognition from Local Scale-Invariant Features. The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999.

[12] Bay, Herbert, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-Up Robust Features (SURF). Computer Vision and Image Understanding Volume 110, Issue 3, June 2008.

[13] Calonder, Michael, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary Robust Independent Elementary Features. ECCV10 Proceedings of the 11th European conference on Computer vision: Part IV 2010.

[14] Calonder, Michael et al. BRIEF: Computing a Local Binary Descriptor Very Fast. Pattern Analysis and Machine Intelligence, Vol. 34, 2012.

[15] Schmidt, Adam, Marek Kraft, Micha Fularz, and Zuzanna Domagaa. The Comparison of Point Feature Detectors and Descriptors in the Context of Robot Navigation. Workshop on Perception for Mobile Robots Autonomy, 2012.

[16] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An Efficient Alternative to SIFT or SURF. ICCV 11 Proceedings of the 2011 International Conference on Computer Vision 2011.

[17] Edward Rosten, Tom Drummond. Machine learning for highspeed corner detection. In European Conference on Computer Vision, volume 1, 2006. I

[18] Rosin, Paul L. Measuring Corner Properties. Computer Vision Image Understanding, Vol.73, No. 2.

[19] Trajkovic, M., and M. Hedley. Fast Corner Detection. Image and Vision Computing Volume 16, Issue 2, 20 February 1998.

[20] Tola, E., V. Lepetit, and P. Fua. A Fast Local Descriptor for Dense Matching. Conference on Computer Vision and Pattern Recognition, 2008.

[21] H. J. Seo and P. Milanfar. Training-free, generic object detection using locally adaptive regression kernels. IEEE Transactions Pattern Analysis and Machine Intelligence, 32(9):16881704, 2010.

[22] Z. Zou, Z. Shi, Y. Guo, and J. Ye, Object detection in 20 years: A survey, arXiv, vol. 1905.05055, 2018.

[23] Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pp. 10971105, 2012.

[24] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, and G. Wang. Recent advances in convolutional neural networks. arXiv preprint arXiv:1512.07108, 2015

[25] R. Girshick, J. Donahue, T. Darrell, and J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580587.

[26] K. He, X. Zhang, S. Ren, and J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, in European conference on computer vision. Springer, 2014, pp. 346361.

[27] R. Girshick, Fast r-cnn, in Proceedings of the IEEE international conference on computer vision, 2015, pp. 14401448.

[28] S. Ren, K. He, R. Girshick, and J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, in Advances in neural information processing systems, 2015, pp. 9199.

[29] T.-Y. Lin, P. Dollar, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, Feature pyramid networks for object detection. in CVPR, vol. 1, no. 2, 2017, p. 4.

[30] [23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, Focal loss for dense object detection, IEEE transactions on pattern analysis and machine intelligence, 2018.

[31] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, Selective search for object recognition, International journal of computer vision, vol. 104, no. 2, pp. 154171, 2013.

[32] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graphbased image segmentation. IJCV, 59(2):167181, 2004.

[33] J. Redmon and A. Farhadi, Yolo9000: better, faster, stronger, arXiv preprint, 2017.

[34] J. Redmon and A. Farhadi, Yolov3: An incremental improvement, arXiv preprint arXiv:1804.02767, 2018

[35] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In CVPR, 2015.

[36] N. Tajbakhsh, Convolutional neural networks for medical image analysis: Full training or fine tuning?, IEEE Trans. Med. Imag., vol. 35, no. 5, pp. 1299-1312, May 2016.

[37] Nikolenko, S. I. Synthetic data for deep learning. ArXiv abs/1909.11512. 2019

[38] Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3D models. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015.

[39] E. Bochinski, V. Eiselein, and T. Sikora. Training a convolutional neural network for multi-class object detection using solely virtual world data. In 13th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pages 278285, Aug 2016.

[40] Param S. Rajpura, Ravi S. Hegde, and Hristo Bojinov. Object detection using deep cnns trained on synthetic images. CoRR, abs/1706.06782, 2017.

[41] Ertugrul Bayraktar, Cihat Bora Yigit, and Pinar Boyraz. A hybrid image dataset toward bridging the gap between real and simulation environments for robotics. Machine Vision and Applications, 30:2340, 2018.

[42] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In CVPR, 2016. 2, 3, 4

[43] Farzan Erlik Nowruzi, Prince Kapoor, Dhanvin Kolhatkar, Fahed Al Hassanat, Robert Lagani‘ere, and Julien Rebut. How much real data do we actually need: Analyzing object detection performance using synthetic and real data. ArXiv, abs/1907.07061, 2019.

[44] Hang Su, Xiatian Zhu, and Shaogang Gong. Open Logo Detection Challenge. In Proceedings of the British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018, 2018

[45] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, A survey on deep transfer learning, https://arxiv.org/abs/1808.01974, 2018.

[46] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786):504507, 2006.

[47] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.

[48] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In ECCV, 2014.

[49] Y. Wang, D. Ramanan, and M. Hebert, Growing a brain: Fine-tuning by increasing model capacity, in IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., Jul. 2017, pp. 30293038.

[50] N. Karessli, Z. Akata, B. Schiele, and A. Bulling. Gaze embeddings for zero-shot image classification. in IEEE Computer Vision and Pattern Recognition (CVPR), 2017.

[51] Xian, Y., Lampert, C.H., Schiele, B., Akata, Z.: Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. arXiv preprint arXiv:1707.00600, 2017.

[52] E. Kodirov, T. Xiang, and S. Gong, Semantic autoencoder for zero-shot learning, in CVPR, 2017.

[53] L. Fe-Fei, R. Fergus, P. Perona: A Bayesian approach to unsupervised one-shot learning of object categories. In Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on, IEEE, 2003.

[54] L. Fei-Fei, R. Fergus, P. Perona: One-shot learning of object categories. In Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2006.

[55] B. Lake, R. Salakhutdinov, J. Gross, J. B. Tenenbaum: One shot learning of simple visual concepts. In Proceedings of the 33rd Annual Conference of the Cognitive Science Society, volume 172, 2011.

[56] Lake, Brenden M, Lee, Chia-ying, Glass, James R, and Tenenbaum, Joshua B. Oneshot learning of generative speech concepts. Cognitive Science Society, 2014.

[57] Maas, Andrew and Kemp, Charles. One-shot learning with bayesian networks. Cognitive Science Society, 2009

[58] Wu, Di, Zhu, Fan, and Shao, Ling. One shot learning gesture recognition from rgbd images. In Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on, IEEE, 2012.

[59] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09, 2009.

[60] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. IJCV, pages 303338, 2010.

[61] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft COCO: Common objects in context. In ECCV. 2014

[62] R. Padilla, S. L. Netto and E. A. B. da Silva, "A Survey on Performance Metrics for Object-Detection Algorithms," 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), Niteri, Brazil, pp. 237-242, 2020.

[63] Everingham, M., Van Gool, L., Williams, C.K.I., J. Winn, Zisserman, A. The PASCAL Visual Object Classes (VOC) Challenge. Int J Comput Vis 88, 303338, 2010.

[64] A. Krizhevsky. Learning multiple layers of features from tiny images. Tech Report, 2009.

[65] Lim, Joseph Jaewhan. Transfer learning by borrowing examples for multiclass object detection. Masters thesis, Massachusetts Institute of Technology, 2012.

[66] G. Koch, R. Zemel, R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In ICML Deep Learning workshop, 2015.

[67] C. Finn, P. Abbeel, and S. Levine. Model-agnostic metalearning for fast adaptation of deep networks. In International Conference on Machine Learning, 2017

[68] J. Snell, K. Swersky, and R. S. Zemel. Prototypical networks for few-shot learning. In Neural Information Processing Systems, 2017.

[69] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In International Conference on Learning Representations, 2018.

[70] B. N. Oreshkin, P. Rodriguez, and A. Lacoste. TADAM: Task dependent adaptive metric for improved few-shot learning. ArXiv e-prints, May 2018.

[71] Sun, Q., Liu, Y., Chua, T.S., Schiele, B. Meta-transfer learning for few-shot learning. In Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[72] Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. Science, 350(6266), 1332-1338. 2015.

[73] Vinyals, Oriol, Blundell, Charles, Lillicrap, Timothy, Kavukcuoglu, Koray, and Wierstra, Daan. Matching networks for one shot learning. In Neural Information Processing Systems (NIPS). arXiv preprint arXiv:1606.04080, 2016.

[74] E. Ahmed, M. Jones, T. K. Marks: An improved deep learning architecture for person re-identification. In Computer Vision and Pattern Recognition, Proceedings of IEEE Computer Society Conference on, CVPR. 2015.

[75] Wiggers, K.L., Britto Jr., A.S., Koerich, A.L., Heutte, L., Oliveira, L.E.S. Image retrieval and pattern spotting using siamese neural network, in: Intl Joint Conf Neural Networks, Budapest. pp. 110. 2019.

[76] Z. Zhang, H. Peng, and Q. Wang. Deeper and wider siamese networks for real-time visual tracking. In CVPR, 2019.

[77] Y. Taigman, M. Yang, M. Ranzato, L. Wolf: Deepface: Closing the gap to human-level performance in face verification. In Computer Vision and Pattern Recognition, Proceedings of IEEE Computer Society Conference on, CVPR. 2014.

[78] Y. Kalantidis, LG. Pueyo, M. Trevisiol, R. van Zwol, Y. Avrithis. Scalable Triangulation-based Logo Recognition. In Proceedings of ACM International Conference on Multimedia Retrieval (ICMR 2011), Trento, Italy, April 2011.

[79] S. Romberg, L. G. Pueyo, R. Lienhart, and R. Van Zwol, Scalable logo recognition in real-world images, in ACM International Conference on Multimedia Retrieval (ICMR). ACM, 2011, p. 25.

[80] Simone Bianco, Marco Buzzelli, Davide Mazzini, and Raimondo Schettini. Deep learning for logo recognition. Neurocomputing, 245:2330, 2017.

[81] Alexis Joly and Olivier Buisson. Logo retrieval with a contrario visual query expansion. In ACM International Conference on Multimedia, pages 581584, 2009.

[82] Hang Su, Shaogang Gong, and Xiatian Zhu. Weblogo-2m: Scalable logo detection by deep learning from the web. In Workshop of the IEEE International Conference on Computer Vision, 2017.

[83] Andras Tzk, Christian Herrmann, Daniel Manger, and Jrgen Beyerer. Open set logo detection and retrieval. arXiv preprint arXiv:1710.10891, 2017.

[84] Yuan Liao, Xiaoqing Lu, Chengcui Zhang, Yongtao Wang, and Zhi Tang. Mutual enhancement for detection of multiple logos in sports videos. In IEEE International Conference on Computer Vision, 2017.

[85] Bromley, Jane, Bentz, James W, Bottou, Leon, Guyon, Isabelle, LeCun, Yann, Moore, Cliff, Sackinger, Eduard, Shah, Roopak: Signature verification using a siamese time delay neural network. In International Journal of Pattern Recognition and Artificial Intelligence, 7 (04):669688, 1993.

[86] P. Viola and M. Jones, Rapid object detection using a boosted cascade of simple features, in Computer Vision and Pattern Recognition, CVPR. 2001.

[87] B. Lavi, M. F. Serj, and I. Ullah. Survey on deep learning techniques for person re-identification task. arXiv preprint arXiv:1807.05284, abs/1807.05284, 2018

[88] Raia Hadsell, Sumit Chopra, Yann LeCun: Dimensionality Reduction by Learning an Invariant Mapping. In Computer Vision and Pattern Recognition, Proceedings of IEEE Computer Society Conference on, CVPR. 2006.

[89] K. Simonyan, A. Zisserman: Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.

[90] G. Huang, Z. Liu, K. Q. Weinberger, L. Maaten: Densely connected convolutional networks. In Computer Vision and Pattern Recognition, Proceedings of IEEE Computer Society Conference on, CVPR. 2017.

[91] K. He, X. Zhang, S. Ren, and J. Sun: Deep residual learning for image recognition. In Computer Vision and Pattern Recognition, Proceedings of IEEE Computer Society Conference on, CVPR. IEEE, 2016.

[92] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. arXiv:1409.0575, 2014.

[93] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In European conference on computer vision, pages 850865. Springer, 2016.

[94] Anfeng He, Chong Luo, Xinmei Tian, and Wenjun Zeng. A twofold siamese network for real-time object tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 48344843, 2018.

[95] Hsieh, T.I., Lo, Y.C., Chen, H.T., Liu, T.L. One-Shot Object Detection with Co-Attention and Co-Excitation. In: NeurIPS, 2019.

[96] Xiang Li, Lin Zhang, Yau Pun Chen, Yu-Wing Tai, Chi-Keung Tang. One-Shot Object Detection without Fine-Tuning. ArXiv preprint arXiv:2005.03819.

[97] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, 2018.

[98] Miaobin Cen and Cheolkon Jung. Fully convolutional siamese fusion networks for object tracking. In 2018 IEEE International Conference on Image Processing, ICIP 2018, Athens, Greece, October 7-10, 2018, pages 37183722, 2018

[99] Tengfei Zhang, Yue Zhang, Xian Sun, Hao Sun, Menglong Yan, Xue Yang, and Kun Fu. Comparison network for one-shot conditional object detection. CoRR, abs/1904.02317, 2019.